WEB DEVELOPMENT LAB RECORD

Subject: Web Technologies / HTML & CSS Programming
Course: Computer Science / Information Technology
Academic Year: 2024-25

INDEX

EXPERIMENT 1: Create your class time table using table tag

Aim:

To create a class timetable using HTML table tags with proper structure and formatting.

Theory:

HTML tables are used to display tabular data in rows and columns. The main elements used for creating tables are:

- `<table>` - Defines a table
- `<tr>` - Defines a table row
- `<th>` - Defines a table header cell

- `<td>` - Defines a table data cell
- `colspan` - Spans a cell across multiple columns
- `rowspan` - Spans a cell across multiple rows
- `border` - Adds border to table

Tables are essential for displaying structured data like timetables, student records, financial data, etc.

Algorithm:

1. Create HTML document structure
2. Add `<table>` element with border attribute
3. Create header row using `<tr>` and `<th>` elements
4. Add time slots and days as headers
5. Create data rows for each time period
6. Use colspan/rowspan for breaks and merged cells
7. Add styling for better appearance
8. Save and test in browser

Sample Code Structure:

```
<!DOCTYPE html>
<html>
<head>
    <title>Class Timetable</title>
</head>
<body>
    <h1 align="center">CLASS TIMETABLE</h1>
    <table border="2" align="center" cellpadding="5" cellspacing="2">
        <tr bgcolor="lightblue">
            <th>Day/Period</th>
            <th>9:00-10:00</th>
            <th>10:00-11:00</th>
            <th>11:00-12:00</th>
            <th>12:00-1:00</th>
            <th>1:00-2:00</th>
            <th>2:00-3:00</th>
        </tr>
        <tr>
            <td bgcolor="lightgreen"><b>Monday</b></td>
            <td>Mathematics</td>
            <td>Physics</td>
            <td>Chemistry</td>
            <td rowspan="6"><b>L<br>U<br>N<br>C<br>H</b></td>
            <td>English</td>
            <td>Computer Science</td>
        </tr>
        <!-- Add more rows for other days -->
```

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

Expected Output:

A well-structured timetable displaying:

- Days of the week in rows
- Time periods in columns
- Subject names in appropriate cells
- Lunch break spanning multiple rows
- Proper borders and alignment

Result:

Class timetable created successfully using HTML table tags with proper formatting and structure.

EXPERIMENT 2: Design a Webpage for your college containing description of courses, department, faculties, library etc.

Aim:

To create a comprehensive college webpage using HTML lists, hyperlinks, and anchor tags for navigation.

Theory:

HTML provides various elements for creating organized content and navigation:

Lists:

- `<ul>` - Unordered list (bullets)
- `<ol>` - Ordered list (numbers)
- `<li>` - List item
- `<dl>` - Definition list

Links:

- `<a href="URL">` - External links
- `<a href="#id">` - Internal page navigation
- `<a name="anchor">` - Named anchors

These elements help create well-structured, navigable webpages.

Algorithm:

1. Create HTML document structure
2. Add navigation menu using unordered list
3. Create sections for different college information
4. Use internal links for navigation
5. Add course details using ordered lists
6. Include faculty information
7. Add library and facilities sections
8. Test all navigation links

Sample Code Structure:

```
<!DOCTYPE html>
<html>
<head>
    <title>College Information</title>
</head>
<body>
    <h1 align="center">WELCOME TO ABC COLLEGE</h1>

    <!-- Navigation Menu -->
    <nav>
        <ul>
            <li><a href="#about">About Us</a></li>
            <li><a href="#courses">Courses</a></li>
            <li><a href="#departments">Departments</a></li>
            <li><a href="#faculty">Faculty</a></li>
            <li><a href="#library">Library</a></li>
            <li><a href="#contact">Contact</a></li>
        </ul>
    </nav>

    <!-- About Section -->
    <section id="about">
        <h2>About Our College</h2>
        <p>ABC College is a premier educational institution...</p>
    </section>

    <!-- Courses Section -->
    <section id="courses">
        <h2>Courses Offered</h2>
        <ol>
            <li>Bachelor of Computer Science</li>
            <li>Bachelor of Information Technology</li>
            <li>Master of Computer Applications</li>
        </ol>
    </section>

    <!-- Add other sections -->
```

```
</body>
</html>
```

Expected Output:

A complete college webpage featuring:

- Navigation menu with internal links
- Organized course listings
- Faculty information
- Library and facilities details
- Working anchor navigation

Result:

College webpage created successfully with proper navigation and organized content using lists and links.

EXPERIMENT 3: Create web page using Frames

Aim:

To create a frame-based webpage layout with header, left navigation, right content, and status bar frames with interactive navigation.

Theory:

HTML frames divide the browser window into multiple independent sections. Key elements:

- `<frameset>` - Defines frame structure
- `<frame>` - Individual frame definition
- `rows` - Horizontal division
- `cols` - Vertical division
- `src` - Frame source file
- `name` - Frame identifier
- `target` - Navigation target

Note: Frames are deprecated in HTML5 but still useful for understanding layout concepts.

Algorithm:

1. Create main frameset document
2. Define frame structure (header, left, right, status)
3. Create individual HTML files for each frame
4. Set up navigation in left frame
5. Configure target attribute for content display
6. Add status bar information
7. Test frame navigation

Sample Code Structure:

Main Frame File (index.html):

```
<!DOCTYPE html>
<html>
<head>
    <title>Frame Layout</title>
</head>
<frameset rows="15%,70%,15%">
    <frame name="header" src="header.html">
    <frameset cols="25%,75%">
        <frame name="left" src="navigation.html">
        <frame name="right" src="content.html">
    </frameset>
    <frame name="status" src="status.html">
    <noframes>
        <body>Your browser does not support frames.</body>
    </noframes>
</frameset>
</html>
```

Navigation Frame (navigation.html):

```
<html>
<body bgcolor="#f0f0f0">
    <h3>Navigation</h3>
    <ul>
        <li><a href="home.html" target="right">Home</a></li>
        <li><a href="about.html" target="right">About</a></li>
        <li><a href="services.html" target="right">Services</a></li>
        <li><a href="contact.html" target="right">Contact</a></li>
    </ul>
</body>
</html>
```

Expected Output:

A frame-based layout with:

- Header frame at top
- Left navigation frame
- Right content frame
- Status bar at bottom
- Working navigation between frames

Result:

Frame-based webpage layout created successfully with proper navigation and content display.

EXPERIMENT 4: Create Your Resume using HTML

Aim:

To design a personal resume webpage using HTML text formatting, links, colors, and list elements.

Theory:

HTML provides various text formatting elements:

Text Elements:

- `<h1>` to `<h6>` - Headings
- `<p>` - Paragraphs
- `<b>`, `<strong>` - Bold text
- `<i>`, `<em>` - Italic text
- `<u>` - Underlined text

Styling:

- `color` attribute for text color
- `size` attribute for font size
- `face` attribute for font family

Sample Code Structure:

```
<!DOCTYPE html>
<html>
<head>
    <title>My Resume</title>
</head>
```

```
<body bgcolor="lightcyan">
    <h1 align="center" color="navy">JOHN DOE</h1>
    <hr>

    <h2>Personal Information</h2>
    <p><b>Email:</b> john.doe@email.com</p>
    <p><b>Phone:</b> +1-234-567-8900</p>

    <h2>Education</h2>
    <ul>
        <li>Bachelor of Computer Science (2020-2024)</li>
        <li>Higher Secondary (2018-2020)</li>
    </ul>

    <h2>Skills</h2>
    <ol>
        <li>HTML & CSS</li>
        <li>JavaScript</li>
        <li>Python Programming</li>
    </ol>

    <h2>Projects</h2>
    <p><a href="project1.html">E-commerce Website</a></p>
    <p><a href="project2.html">Student Management System</a></p>
</body>
</html>
```

Result:

Professional resume webpage created with proper formatting, colors, and organized content.

EXPERIMENT 5: Create a Web Page of a Super Market using Internal CSS

Aim:

To develop a supermarket website layout using HTML with internal CSS styling.

Theory:

Internal CSS is defined within `<style>` tags in the `<head>` section:

Syntax:

```
<head>
    <style>
        selector {
            property: value;
            property: value;
        }
    </style>
</head>
```

Common Properties:

- `color` - Text color
- `background-color` - Background color
- `font-size` - Font size
- `margin`, `padding` - Spacing
- `border` - Element borders

Sample Code Structure:

```
<!DOCTYPE html>
<html>
<head>
    <title>Super Market</title>
    <style>
        body {
            background-color: #f5f5f5;
            font-family: Arial, sans-serif;
        }
        .header {
            background-color: #2c3e50;
            color: white;
            padding: 20px;
            text-align: center;
        }
        .product-grid {
            display: grid;
            grid-template-columns: repeat(3, 1fr);
            gap: 20px;
            padding: 20px;
        }
        .product-card {
            background-color: white;
            border: 1px solid #ddd;
            padding: 15px;
            text-align: center;
        }
    </style>
</head>
<body>
    <div class="header">
        <h1>Fresh Mart Supermarket</h1>
    </div>

    <div class="product-grid">
        <div class="product-card">
            <h3>Fresh Fruits</h3>
            <p>Price: $2.99/kg</p>
        </div>
        <!-- Add more products -->
    </div>
```

```
</body>
</html>
```

Result:

Attractive supermarket webpage created with internal CSS styling for layout and design.

EXPERIMENT 6: Use Inline CSS to format your resume

Aim:

To apply inline CSS styling to the previously created resume for enhanced presentation.

Theory:

Inline CSS is applied directly to HTML elements using the `style` attribute:

Syntax: `<element style="property: value; property: value;">`

Characteristics:

- Highest specificity
- Overrides external and internal CSS
- Applied to individual elements
- Not reusable

Sample Code Structure:

```
<h1 style="color: #2c3e50; text-align: center; font-family: Georgia;">JOHN DOE</h1>
<p style="background-color: #ecf0f1; padding: 10px; border-left: 4px solid #3498db;">
    <b style="color: #e74c3c;">Email:</b> john.doe@email.com
</p>
```

Result:

Resume enhanced with inline CSS styling for improved visual presentation.

EXPERIMENT 7: Use External CSS to format your time table

Aim:

To create and link an external CSS file to style the class timetable.

Theory:

External CSS is stored in separate .css files and linked using:

```
<link rel="stylesheet" href="filename.css">
```

Advantages:

- Reusable across multiple pages
- Better organization
- Smaller HTML file size
- Easier maintenance

Sample Implementation:

HTML File:

```
<head>
    <link rel="stylesheet" href="timetable.css">
</head>
```

CSS File (timetable.css):

```
table {
    border-collapse: collapse;
    width: 80%;
    margin: 20px auto;
}

th, td {
    border: 2px solid #2c3e50;
    padding: 12px;
    text-align: center;
}

th {
    background-color: #3498db;
    color: white;
}
```

Result:

Timetable styled using external CSS with professional appearance and reusable styling.

EXPERIMENT 8: Use all CSS types to format college web page

Aim:

To demonstrate the combination of all three CSS types in a single webpage with proper precedence understanding.

Theory:

CSS Specificity Hierarchy:

1. Inline CSS (Highest priority)
2. Internal CSS
3. External CSS (Lowest priority)

When multiple styles apply to the same element, the one with higher specificity takes precedence.

Sample Implementation:

```
<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" href="external.css">
    <style>
        /* Internal CSS */
        h1 { color: blue; }
        .highlight { background-color: yellow; }
    </style>
</head>
<body>
    <h1 style="color: red;">College Name</h1> <!-- Inline CSS wins -->
    <p class="highlight">This paragraph has internal CSS styling.</p>
</body>
</html>
```

Result:

College webpage demonstrating all CSS types with proper understanding of specificity and cascading.

EXPERIMENT 9: Create college website for mobile device

Aim:

To create a responsive college website optimized for mobile devices using viewport meta tag and responsive design principles.

Theory:

Mobile Responsive Design Elements:

- Viewport meta tag: `<meta name="viewport" content="width=device-width, initial-scale=1">`
- Media queries for different screen sizes
- Flexible layouts using CSS Grid/Flexbox
- Scalable images and text

Sample Code Structure:

```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <style>
        @media screen and (max-width: 600px) {
            .container { width: 100%; }
            .nav-menu { flex-direction: column; }
        }
    </style>
</head>
<body>
    <div class="container">
        <!-- Mobile-optimized content -->
    </div>
</body>
</html>
```

Result:

Responsive college website created successfully with mobile optimization and viewport configuration.

LAB ASSESSMENT CRITERIA

Component	Marks	Description
Code Quality	30	Proper syntax, structure, and organization
Output	25	Correct functionality and appearance
Documentation	20	Complete aim, theory, and explanations
Innovation	15	Creative design and additional features

Component	Marks	Description
Viva	10	Understanding of concepts
Total	**100**	

CONCLUSION

This lab record demonstrates comprehensive understanding of HTML and CSS concepts including:

- HTML table creation and formatting
- List structures and navigation
- Frame-based layouts (legacy)
- Resume design with HTML
- All three CSS implementation methods
- Responsive web design principles

These experiments provide a solid foundation for modern web development practices.

Student Name: _____
Roll Number: _____
Date: _____
Instructor Signature: _____