Programme 1 - Design a simple static HTML web page using:

- Ordered lists
- Unordered lists
- Tables

Display and maintain student record information in a structured format.

Aim

To design and develop a simple static HTML web page that displays and maintains student record information using **ordered lists**, **unordered lists**, and **tables**.

Algorithm / Steps

1. Start the program.
2. Open any text editor (such as Notepad or VS Code).
3. Create a new file and save it with the extension .html.
4. Begin the HTML document using the <!DOCTYPE html> declaration.
5. Create the root structure of the web page using <html>, <head>, and <body> tags.
6. Inside the <head> section:
    - Specify the title of the web page using the <title> tag.
7. Inside the <body> section:
    - Display a suitable heading using heading tags (<h1>, <h2>, etc.).
8. Use an **unordered list (<ul>)** to display general student details such as:
    - Name
    - Department
    - Year
    - Section
9. Use an **ordered list (<ol>)** to display academic or procedural information such as:
    - Semester details
    - Course sequence
    - Steps for registration

10. Create a **table (<table>)** to display student records in a structured format:
    - Define table rows using <tr>.
    - Define table headings using <th>.
    - Define table data using <td>.
11. Add appropriate attributes to the table such as:
    - Border
    - Cell padding
    - Cell spacing (if required)
12. Insert multiple student records into the table.
13. Save the HTML file.
14. Open the file in any web browser to view the output using Live server in VS code.
15. Verify that the ordered list, unordered list, and table are displayed correctly.
16. Apply CSS properties to make the html page colorfull.
17. Stop the program.

Programme 2 - **Design static web pages** for an online book store website:

- **Home Page:** Static home page with three frames
- **Login Page**
- **Catalogue Page:** Display all available books in a table
- **Registration Page**

Aim

To design and develop **static web pages** for an **Online Book Store website** using HTML, including:

- Home page with three frames
- Login page
- Catalogue page displaying books in a table
- Registration page for new users

Algorithm

1. Start the program.
2. Create the **Home Page** using HTML.
3. Divide the Home Page into **three frames**:
    - Header frame
    - Navigation frame
    - Content frame
4. Add navigation links in the Home Page to:
    - Login Page
    - Catalogue Page
    - Registration Page
5. Design the **Login Page** using an HTML form with username and password fields.
6. Design the **Catalogue Page** using an HTML table to display available books with details such as book name, author, price, and category.
7. Design the **Registration Page** using an HTML form to collect user details like name, email, password, gender, and address.
8. Link all pages properly using anchor (<a>) tags.
9. Save all HTML files and test them in a web browser.
10. Stop the program.

Steps to Develop the Program

Step 1: Create the Home Page

- Create an HTML file named **index.html.**
- Use frames to divide the page into three sections:
    - Top frame for website title
    - Left frame for navigation menu
    - Right frame for displaying pages
- Create a anchor link for login page, catalogue page and Registration page.

Step 2: Design the Login Page

- Create a file named login.html and link this to index page.
- Use <form> tag.
- Include:
    - Username field
    - Password field
    - Login button

Step 3: Design the Catalogue Page

- Create a file named catalogue.html and link this to index page.
- Use <table> tag.
- Display book details such as:
    - Book ID
    - Book Name
    - Author
    - Price

Step 4: Design the Registration Page

- Create a file named register.html and link this to index page.
- Use <form> tag.
- Collect user information:
    - Name
    - Email
    - Password
    - Gender
    - Address
- Include submit and reset buttons.

Step 5: Link All Pages

- Use anchor (<a>) tags in the navigation frame in the index page.
- Ensure all pages open correctly within the content frame.

Step 6: Execute and Verify

- Open index.html in a web browser.
- Verify navigation and page layout.
- Ensure all pages load correctly.

Result

Thus, static web pages for an **Online Book Store website** were successfully designed using HTML.

Programme 3 - **Design an HTML web page** using anchor (<a>) tags and an iframe to display linked content within the same page.

The iframe should load a web page containing an image using the <img> tag.

Aim

To design and develop an HTML web page using **anchor (<a>) tags** and an **iframe** to display linked content within the same page, where the iframe loads a web page containing an image using the <img> tag.

Algorithm / Steps

1. Start the program.
2. Open a text editor such as Notepad or Visual Studio Code.
3. Create a new project folder.
4. Create the following HTML files:
    - index.html (Main Page)
    - imagepage.html (Web page containing image)

Main Page (Anchor Tags and Iframe)

5. Open index.html and begin the HTML document using the <!DOCTYPE html> declaration.
6. Create the basic structure using <html>, <head>, and <body> tags.
7. Add a suitable heading to the page using heading tags.
8. Use **anchor (<a>) tags** to create links that target the iframe.
9. Assign a name or id to the iframe to enable loading of linked content.
10. Insert an **iframe (<iframe>)** element in the web page to display the linked content within the same page.
11. Set appropriate attributes for the iframe such as:
    - Width
    - Height
    - Border (optional)
12. Save the index.html file.

Image Page (Image Display)

13. Open imagepage.html.
14. Create the HTML document structure.
15. Insert an image using the **<img> tag**.
16. Specify the image source using the src attribute.
17. Set appropriate attributes for the image such as:
    - Width
    - Height
    - Alternate text (alt)
18. Save the imagepage.html file.

Execution

19. Open index.html in a web browser.

20. Click on the anchor link.
21. Verify that:
    - The linked content loads inside the iframe.
    - The iframe displays the web page containing the image.
22. Stop the program.

Result

Thus, an HTML web page using anchor tags and an iframe to display linked content within the same page, where the iframe loads a web page containing an image using the <img> tag, is successfully designed and executed.

Programme 4 - **Create a web page** using HTML that contains a form to collect student information including:

- Name
- Email
- Branch
- Gender
- Address

Aim

To design and develop an HTML web page containing a **form** to collect student information such as **Name, Email, Branch, Gender, and Address**, using proper form settings and input elements.

Algorithm / Steps

Step 1: File Creation

1. Start the program.

2. Open a text editor such as Notepad or Visual Studio Code.
3. Create a new HTML file and save it as student_form.html.

Step 2: Basic HTML Structure

4. Begin the HTML document using the <!DOCTYPE html> declaration.
5. Create the basic structure using:
    - <html>
    - <head>
    - <body>
6. Inside the <head> section:
    - Set the title of the web page using the <title> tag.
7. Inside the <body> section:
    - Display a suitable heading using heading tags.

Step 3: Form Configuration (Complete Settings)

8. Create a form using the <form> tag.
9. Set the **action attribute**:
    - Specifies the destination page or server where the form data is sent.
    - Example: action="submit.html"
10. Set the **method attribute**:
    - Use POST to securely send form data.
    - Prevents sensitive data from appearing in the URL.
11. Assign a **name or id attribute** to uniquely identify the form.

Step 4: Input Field – Name

12. Add a text input field using the <input> tag.
13. Set the following attributes:
    - type="text" – accepts textual data

- name="student_name" – identifies the data
- placeholder="Enter your name" – guides the user
- required – makes the field mandatory

Step 5: Input Field – Email

14. Add an email input field.
15. Set the attributes:
    - type="email" – validates email format automatically
    - name="email"
    - placeholder="Enter your email"
    - required

Step 6: Input Field – Branch

16. Use a dropdown list to select branch using the <select> tag.
17. Define multiple <option> values such as:
    - CSE
    - ECE
    - EEE
    - MECH
18. Assign:
    - name="branch"
    - required

Step 7: Input Field – Gender

19. Use radio buttons to select gender.
20. Create radio inputs using:
    - type="radio"
    - Same name="gender" for grouping

- Different value attributes

21. Make selection mandatory using required.

Step 8: Input Field – Address

22. Use the <textarea> tag to collect address details.
23. Set attributes:
    - name="address"
    - rows and cols for size
    - placeholder="Enter your address"
    - required

Step 9: Form Buttons

24. Add a **Submit button** using:
    - <input type="submit">
    - Sends form data to the action page.
25. Add a **Reset button** using:
    - <input type="reset">
    - Clears all entered data.

Step 10: Execution

26. Save the HTML file.
27. Open the file in any web browser.
28. Enter student details in all fields.
29. Click the Submit button and verify that:
    - Mandatory validation works correctly.
    - Data is submitted successfully and page is transferred to submit.html.
30. Stop the program.

Result

Thus, an HTML web page containing a form to collect student information including Name, Email, Branch, Gender, and Address using proper form settings is successfully designed and executed.

Programme 5 - **Create a web page** to embed a map along with:

- Hot spots
- Frames
- Links

Aim

To design and develop a web page that embeds a **map** and provides **hot spots**, **frames**, and **hyperlinks** to navigate and display related content.

Algorithm / Steps

Step 1: File Creation

1. Start the program.
2. Open a text editor such as Notepad or Visual Studio Code.
3. Create a new project folder.
4. Create the following HTML files:
    - index.html – Main page with frames
    - map.html – Page containing embedded map and hot spots
    - content.html – Page to display linked information

Step 2: Create the Main Page with Frames

5. Open index.html.

6. Start the HTML document using the <!DOCTYPE html> declaration.
7. Create a **frameset** to divide the browser window into multiple frames:
    - One frame for navigation links
    - One frame for displaying map and content
8. Assign names to each frame to enable link targeting.
9. Save the file.

Step 3: Navigation Links

10. In the navigation frame, create hyperlinks using **anchor (<a>) tags**.
11. Use the href attribute to specify target pages.
12. Use the target attribute to load content in the appropriate frame.
13. Save the navigation frame content.

Step 4: Embed the Map

14. Open map.html.
15. Create the basic HTML structure.
16. Embed a map using:
    - An embedded map service (such as Google Maps iframe), or
    - An image map using the <img> tag.
17. Set appropriate attributes such as:
    - Width
    - Height
    - Border

Step 5: Create Hot Spots

18. Use an **image map** to define hot spots on the map.
19. Define a <map> element and assign a name.
20. Create clickable regions using <area> tags.

21. Specify the following attributes for each hot spot:
    - shape (rectangle, circle, polygon)
    - coords (coordinates of the region)
    - href (linked page)
    - target (frame where content is displayed)
22. Save the map file.

Step 6: Content Page

23. Open content.html.
24. Create a simple HTML page with text or images related to the selected hot spot.
25. Save the file.

Step 7: Execution

26. Open index.html in a web browser.
27. Click the navigation links and verify that:
    - Pages load in the correct frames.
28. Click on different hot spots on the map.
29. Verify that:
    - Clicking a hot spot loads related content.
    - Links function correctly within frames.
30. Stop the program.

Result

Thus, a web page embedding a map using hot spots, frames, and hyperlinks is successfully designed and executed.