Ex. No. : **Date :**

Roll No. : **Reg. No. :**

HTML – Element - Tables

Aim:

Program to design a class timetable using HTML basic elements.

Procedure:

1. Tables are defined with the **table** element
2. Use the ***border*** attribute specifies the table's border width in pixels. To create a table without a border, set **border** to **"0"**.
3. Use the ***tr*** element to define an individual *table row*.
4. The columns in the head section are defined with **th** elements.
5. *Data cells* contain individual pieces of data and are defined with **td** (*table data*) elements within each row.
6. Table cells are sized to fit the data they contain. Document authors can create larger data cells by using attributes ***rowspan*** and ***colspan***. The values assigned to these attributes specify the number of rows or columns occupied by a cell.
7. Use the attribute **rowspan = "2"** to allow the cell to use two vertically adjacent cells (thus the cell *spans* two rows).
8. Use the attribute **colspan = "4"** to widen the header cell to span four cells.

Ex. No. : **Date :**

Roll No. : **Reg. No. :**

HTML – Form Elements

Aim:

Program to design a email registration form using HTML Form elements.

Procedure:

1. The form is defined by a ***form*** element.
   ```
   <form method = "post" action = "/cgi-bin/formmail">
   ```
2. Use thea attribute ***method*** specifies how the form's data is sent to the Web server.Using ***method = "post"*** appends form data to the browser request, which contains the protocol (i.e., HTTP) and the requested resource's URL. Scripts located on the Web server's computer (or on a computer accessible through the network) can access the form data sent as part of the request. For example, a script may take the form information and update an electronic mailing list. The other possible value, ***method = "get"*** appends the form data directly to the end of the URL.
3. The ***action*** attribute in the **<form>** tag specifies the URL of a script on the Web server; in this case, it specifies a script that e-mails form data to an address. Most Internet Service Providers (ISPs) have a script like this on their site; ask the Web site system administrator how to set up an XHTML document to use the script correctly.
4. Use the **type** of **input** as ***"text"*** **input** inserts a *text box* into the form. Users can type data in text boxes.
5. The **input** element's ***size*** attribute specifies the number of characters visible in the text box. Optional attribute ***maxlength*** limits the number of characters input into the text box.
6. There are two types of **input** elements in lines
   ```
   <input type = "submit" value = "Submit Your Entries" />
   <input type = "reset" value = "Clear Your Entries" />
   ```
7. The ***"submit"*** **input** element is a button. When the user presses a **"submit"** button, the browser sends the data in the form to the Web server for processing. The ***value*** *attribute* sets the text displayed on the button (the default value is **Submit Query**).
8. The ***"reset"*** **input** element allows a user to reset all **form** elements to their default values. The **value** attribute of the **"reset" input** element sets the text displayed on the button (the default value is **Reset**).
9. The ***textarea*** element inserts a multiline text box, called a *text area*, into the form. The number of rows is specified with the ***rows*** *attribute* and the number of columns (i.e., characters) is specified with the ***cols*** *attribute.* In this example, the **textarea** is four rows high and 36 characters wide. To display default text in the text area, place the text between the **<textarea>** and **</textarea>** tags. Default text can be specified in other **input** types, such as text boxes, by using the **value** attribute.
10. The ***"password"*** input in lines inserts a password box with the specified **size**. A password box allows users to enter sensitive information, such as credit card numbers and passwords, by "masking" the information input with asterisks. The actual value input is sent to the Web server, not the character that mask the input.

11. Checkboxes enable users to select from a set of options. When a user selects a checkbox, a check mark appears in the check box. Otherwise, the checkbox remains empty. Each ***"checkbox"*** **input** creates a new checkbox. Checkboxes can be used individually or in groups. Checkboxes that belong to a group are assigned the same **name**.
12. Radio buttons are similar to checkboxes, except that only one radio button in a group of radio buttons may be selected at any time. The radio buttons in a group have the same **name** attributes and are distinguished by their different **value** attributes. The attribute-value pair ***checked = "checked"*** indicates which radio button, if any, is selected initially. The **checked** attribute also applies to checkboxes.
13. The ***select*** element provides a drop-down list of items from which the user can select an item. The **name** attribute identifies the drop-down list. The ***option*** element adds items to the drop-down list. The **option** element's ***selected*** *attribute* specifies which item initially is displayed as the selected item in the **select** element.

Ex. No. : **Date :**

Roll No. : **Reg. No. :**

XHTML - Framesets

Aim:

Program to design web pages using basic elements, hyperlinks and to perform web navigation using XHTML.

Procedure:

1. *frames* allows the browser to display more than one XHTML document simultaneously.
2. Use the <***frameset***> tag **`<frameset cols = "110,*">`** which informs the browser that the page contains frames.
3. Use attribute ***cols*** to specify the frameset's column layout. The value of **cols** gives the width of each frame, either in pixels or as a percentage of the browser width.
4. The attribute **cols = "110,*"** informs the browser that there are two vertical frames. The first frame extends **110** pixels from the left edge of the browser window and the second frame fills the remainder of the browser width (as indicated by the asterisk).
5. The **frameset** attribute ***rows*** can be used to specify the number of rows and the size of each row in a frameset.
6. The documents that will be loaded into the **frameset** are specified with ***frame*** elements.
7. The Attribute **src** specifies the URL of the page to display in the frame. Each frame has **name** and **src** attributes. The first frame (which covers **110** pixels on the left side of the **frameset**) is named **leftframe** and displays the page **nav.html**. The second frame is named main and displays the page **main.html**.
8. The Attribute **name** identifies a frame, enabling hyperlinks in a **frameset** to specify the ***target* frame** in which a linked document should display when the user clicks the link.
9. The Anchor attribute **target** specifies that the linked documents are loaded in frame **main**. A **target** can be set to a number of preset values: **"_blank"** loads the page into a new browser window, **"_self"** loads the page into the frame in which the anchor element appears and **"_top"** loads the page into the full browser window (i.e., removes the **frameset**).
10. Use the **frameset** element to create more complex layouts in a Web page by nesting **framesets**.

Ex. No. : **Date :**

Roll No. : **Reg. No. :**

CSS

Aim:

Program to design web pages using basic elements, hyperlinks and to perform web navigation using CSS.

Procedure:

Inline Style Sheets

1. Create inline styles that declare an individual element's format using attribute style.
2. Apply inline styles to p elements to alter their font size and color.
3. Use the attribute style to specify the style for an element.
4. Create CSS property (the font-size property) followed by a colon and a value.
5. Use the two properties, font-size and color, separated by a semicolon.

Embedded Style Sheets

1. Use the style element to define the embedded style sheet.
2. Place the Styles in the head to apply matching elements in the entire document, not just to a single element.
3. Use the type attribute to specify specifies the Multipurpose Internet Mail Extension (MIME) type that describes a file's content. CSS documents use the MIME type text/css.
4. Use the body of the style sheet to declare the CSS rules for the style sheet.
5. The body of each rule is enclosed in curly braces ({ and }).
6. Declare a style class. Class declarations are preceded with a period and are applied to elements only of that class.
7. Use the property name is followed by a colon (:) and the value of that property. Multiple properties are separated by semicolons (;).

Linking External Style Sheets

1. Create a link element, which uses the rel attribute to specify a relationship between the current document and another document.
2. Declare the linked document to be a stylesheet for this document.
3. Use the type attribute to specify the MIME type as text/css.
4. Use the href attribute provides the URL for the document containing the style sheet.

Ex. No. : **Date :**

Roll No. : **Reg. No. :**

Image Maps

Aim:

Program to create and use image maps:

i. To embed a map in a web page.
ii. To fix the hot spots in that map.
iii. Show all the related information when the hot spots are clicked.

Procedure:

1. Define an image maps by using a map element.
2. Use attribute id to identify the image map.
3. Define hotspots with area elements.
4. Use attribute href to specify the link's target (i.e.,the resource to which to link).
5. Use attributes shape and coords to specify the hotspot's shape and coordinates, respectively.
6. Use attribute alt to provides alternate text for the link.
7. Use the markup to create a rectangular hotspot (shape = "rect") for the coordinates specified in the coords attribute (For rectangular hotspots, the required coordinates are those of the upper-left and lower-right corners of the rectangle).
8. Use the map area to assign the shape attribute "poly" to create a hotspot in the shape of a polygon using the coordinates in attribute coords (These coordinates represent each vertex, or corner, of the polygon).
9. Use the map area to assign the shape attribute "circle" to create a circular hotspot (the coords attribute specifies the circle's center coordinates and the circle's radius, in pixels).
10. Use an image map with an img element, the img element's usemap attribute is assigned the id of a map.
11. Locate the image map within the same document so internal linking is used.

Ex. No. : **Date :**

Roll No. : **Reg. No. :**

Form Validation – User Registration Form

Aim:

Program to create a User Registration form with First Name, Last name, Address, City, State, Country, Pincode, Username and Password fields for a General login webpage and satisfy the following criteria:

To create a validate ( ) function that does the following:

(a) Checks that the First Name, Last Name, City, Country, Username, and Password fields are filled out.
(b) Checks that the Pincode is exactly 6 numeric's.
(c) Checks that the state is exactly two characters.
(d) Checks that the email is a valid email address.

- false if email has fewer than 6 characters
- false if email does not contain an @ symbol
- false if email does not contain a period (.)
- true otherwise

Procedure:

1. The form is defined by a ***form*** element.
   `<form method = "post" action = "/cgi-bin/formmail">`
2. Use thea attribute ***method*** specifies how the form's data is sent to the Web server.Using ***method*** = ***"post"*** appends form data to the browser request, which contains the protocol (i.e., HTTP) and the requested resource's URL. Scripts located on the Web server's computer (or on a computer accessible through the network) can access the form data sent as part of the request. For example, a script may take the form information and update an electronic mailing list. The other possible value, ***method*** = ***"get"*** appends the form data directly to the end of the URL.
3. The ***action*** attribute in the **<form>** tag specifies the URL of a script on the Web server; in this case, it specifies a script that e-mails form data to an address. Most Internet Service Providers (ISPs) have a script like this on their site; ask the Web site system administrator how to set up an XHTML document to use the script correctly.
4. Use the **type** of **input** as ***"text"*** **input** inserts a *text box* into the form. Users can type data in text boxes.
5. The **input** element's ***size*** attribute specifies the number of characters visible in the text box. Optional attribute ***maxlength*** limits the number of characters input into the text box.
6. There are two types of **input** elements in lines
   `<input type = "submit" value = "Submit Your Entries" />`
   `<input type = "reset" value = "Clear Your Entries" />`
7. The ***"submit"*** **input** element is a button. When the user presses a **"submit"** button, the browser sends the data in the form to the Web server for processing. The ***value*** *attribute* sets the text displayed on the button (the default value is **Submit Query**).

8. The ***"reset"*** **input** element allows a user to reset all **form** elements to their default values. The **value** attribute of the **"reset" input** element sets the text displayed on the button (the default value is **Reset**).
9. The ***textarea*** element inserts a multiline text box, called a *text area*, into the form. The number of rows is specified with the ***rows*** *attribute* and the number of columns (i.e., characters) is specified with the ***cols*** *attribute.* In this example, the **textarea** is four rows high and 36 characters wide. To display default text in the text area, place the text between the **<textarea>** and **</textarea>** tags. Default text can be specified in other **input** types, such as text boxes, by using the **value** attribute.
10. The ***"password"*** input in lines inserts a password box with the specified **size**. A password box allows users to enter sensitive information, such as credit card numbers and passwords, by "masking" the information input with asterisks. The actual value input is sent to the Web server, not the character that mask the input.
11. Checkboxes enable users to select from a set of options. When a user selects a checkbox, a check mark appears in the check box. Otherwise, the checkbox remains empty. Each ***"checkbox"*** **input** creates a new checkbox. Checkboxes can be used individually or in groups. Checkboxes that belong to a group are assigned the same **name**.
12. Radio buttons are similar to checkboxes, except that only one radio button in a group of radio buttons may be selected at any time. The radio buttons in a group have the same **name** attributes and are distinguished by their different **value** attributes. The attribute-value pair ***checked = "checked"*** indicates which radio button, if any, is selected initially. The **checked** attribute also applies to checkboxes.
13. The ***select*** element provides a drop-down list of items from which the user can select an item. The **name** attribute identifies the drop-down list. The ***option*** element adds items to the drop-down list. The **option** element's ***selected*** *attribute* specifies which item initially is displayed as the selected item in the **select** element.
14. Use the events for processing forms - onsubmit and onreset.
15. These events fire when a form is submitted or reset, respectively.

Ex. No. : **Date :**

Roll No. : **Reg. No. :**

Form Validation – Job Registration Form

Aim:

Program to design a Job Registration form / Sign Up form using HTML. The Form contains Text box [name], Radio button [sex], Text area [comments], List Items [country], File [Browse], Label Field, Password field [password] and check box.

Procedure:

1. The form is defined by a ***form*** element.
   ```
   <form method = "post" action = "/cgi-bin/formmail">
   ```
2. Use thea attribute ***method*** specifies how the form's data is sent to the Web server.Using ***method = "post"*** appends form data to the browser request, which contains the protocol (i.e., HTTP) and the requested resource's URL. Scripts located on the Web server's computer (or on a computer accessible through the network) can access the form data sent as part of the request. For example, a script may take the form information and update an electronic mailing list. The other possible value, ***method = "get"*** appends the form data directly to the end of the URL.
3. The ***action*** attribute in the **<form>** tag specifies the URL of a script on the Web server; in this case, it specifies a script that e-mails form data to an address. Most Internet Service Providers (ISPs) have a script like this on their site; ask the Web site system administrator how to set up an XHTML document to use the script correctly.
4. Use the **type** of **input** as ***"text"*** **input** inserts a *text box* into the form. Users can type data in text boxes.
5. The **input** element's ***size*** attribute specifies the number of characters visible in the text box. Optional attribute ***maxlength*** limits the number of characters input into the text box.
6. There are two types of **input** elements in lines
   ```
   <input type = "submit" value = "Submit Your Entries" />
   <input type = "reset" value = "Clear Your Entries" />
   ```
7. The ***"submit"*** **input** element is a button. When the user presses a **"submit"** button, the browser sends the data in the form to the Web server for processing. The ***value*** *attribute* sets the text displayed on the button (the default value is **Submit Query**).
8. The ***"reset"*** **input** element allows a user to reset all **form** elements to their default values. The **value** attribute of the **"reset" input** element sets the text displayed on the button (the default value is **Reset**).
9. The ***textarea*** element inserts a multiline text box, called a *text area*, into the form. The number of rows is specified with the ***rows*** *attribute* and the number of columns (i.e., characters) is specified with the ***cols*** *attribute.* In this example, the **textarea** is four rows high and 36 characters wide. To display default text in the text area, place the text between the **<textarea>** and **</textarea>** tags. Default text can be specified in other **input** types, such as text boxes, by using the **value** attribute.

10. The ***"password"*** input in lines inserts a password box with the specified **size**. A password box allows users to enter sensitive information, such as credit card numbers and passwords, by “masking” the information input with asterisks. The actual value input is sent to the Web server, not the character that mask the input.
11. Checkboxes enable users to select from a set of options. When a user selects a checkbox, a check mark appears in the check box. Otherwise, the checkbox remains empty. Each ***"checkbox"*** **input** creates a new checkbox. Checkboxes can be used individually or in groups. Checkboxes that belong to a group are assigned the same **name**.
12. Radio buttons are similar to checkboxes, except that only one radio button in a group of radio buttons may be selected at any time. The radio buttons in a group have the same **name** attributes and are distinguished by their different **value** attributes. The attribute-value pair ***checked = "checked"*** indicates which radio button, if any, is selected initially. The **checked** attribute also applies to checkboxes.
13. The ***select*** element provides a drop-down list of items from which the user can select an item. The **name** attribute identifies the drop-down list. The ***option*** element adds items to the drop-down list. The **option** element’s ***selected*** *attribute* specifies which item initially is displayed as the selected item in the **select** element.
14. Use the events for processing forms - onsubmit and onreset.
15. These events fire when a form is submitted or reset, respectively.