1.Lists, Links and Images

a) Write a HTML program, to explain the working of lists. Note: It should have an ordered list, unordered list, nested lists and ordered list in an unordered list and definition lists.

A) HTML Lists:

HTML lists allow web developers to group a set of related items in lists**.**

- **Ordered List:**
  An ordered list is used when the order of items is important.

An ordered list starts with the <ol> tag. Each list item starts with the <li> tag. Each list item is numbered automatically by the browser. It is commonly used for steps in a process, ranking, or sequential instructions.

- **Unordered List:**
  An unordered list is used when the order of items is not important. An unordered list starts with the <ul> tag. Each list item starts with the <li> tag. The items are usually displayed with bullet points by default. It is ideal for listing features, options, or items that don't require a specific order.
- **Nested List:**
  A nested list occurs when one list is placed inside another list. This is useful when you want to categorize or organize information into subcategories.
- **Ordered List in an Unordered List:**
  This is when an ordered list is placed inside an unordered list, typically used when you have groups of items where each group has its own order, but the groups themselves do not need to be ordered.
- **Definition Lists:**
  A description list is a list of terms, with a description of each term. The <dl> tag defines the description list, the <dt> tag defines the term (name), and the <dd> tag describes each term. It is often used for glossaries or FAQs.

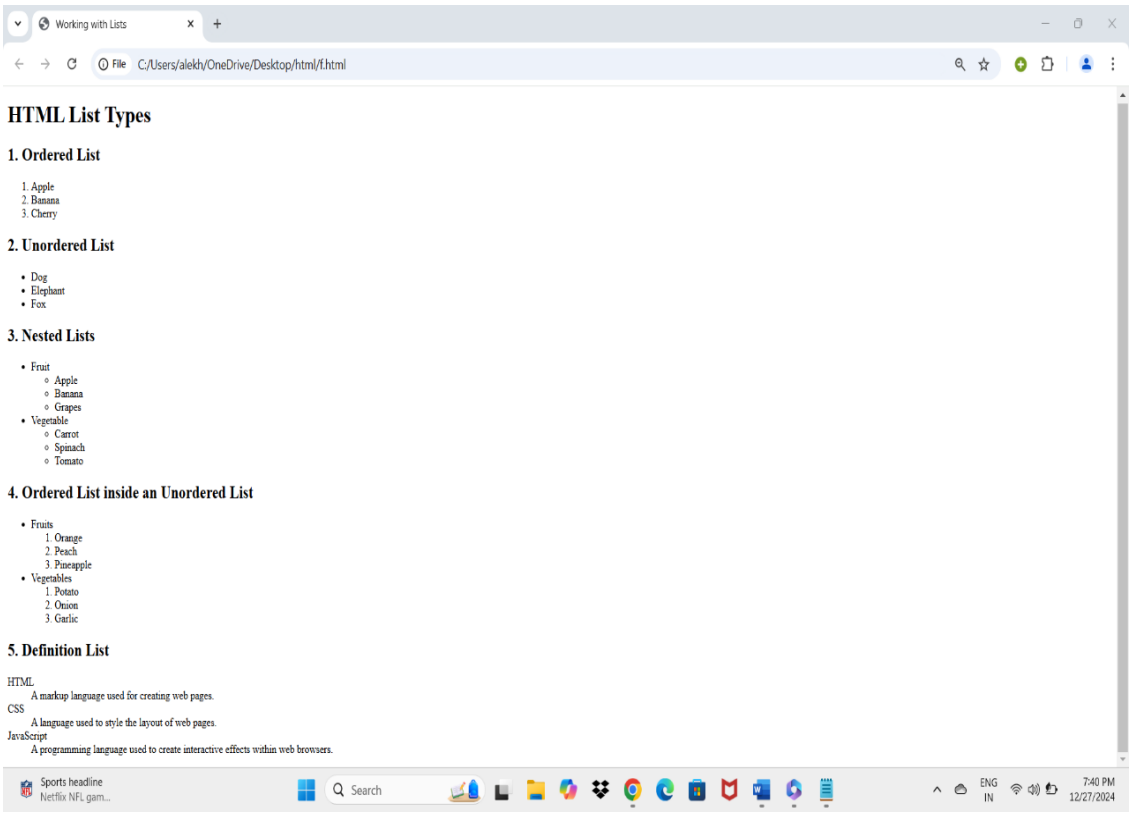
HTML Code:

```
<!DOCTYPE html>
<html>
<head>
    <title>Working with Lists</title>
</head>
<body>
    <h1>HTML List Types</h1>
    <h2>1. Ordered List</h2>
    <ol>
        <li>Apple</li>
        <li>Banana</li>
        <li>Cherry</li>
    </ol>
    <h2>2. Unordered List</h2>
    <ul>
        <li>Dog</li>
        <li>Elephant</li>
        <li>Fox</li>
    </ul>
    <h2>3. Nested Lists</h2>
    <ul>
        <li>Fruit
            <ul>
                <li>Apple</li>
                <li>Banana</li>
                <li>Grapes</li>
```

```
        </ul>
    </li>
    <li>Vegetable
        <ul>
            <li>Carrot</li>
            <li>Spinach</li>
            <li>Tomato</li>
        </ul>
    </li>
</ul>
<h2>4. Ordered List inside an Unordered List</h2>
<ul>
    <li>Fruits
        <ol>
            <li>Orange</li>
            <li>Peach</li>
            <li>Pineapple</li>
        </ol>
    </li>
    <li>Vegetables
        <ol>
            <li>Potato</li>
            <li>Onion</li>
            <li>Garlic</li>
        </ol>
    </li>
</ul>
<h2>5. Definition List</h2>
```

```
    <dl>
        <dt>HTML</dt>
        <dd>A markup language used for creating web pages.</dd>
        <dt>CSS</dt>
        <dd>A language used to style the layout of web pages.</dd>
        <dt>JavaScript</dt>
        <dd>A programming language used to create interactive effects within web browsers.</dd>
    </dl>
</body>
</html>
```

Output:



b) Write a HTML program, to explain the working of hyperlinks using tag and href, target Attributes.

A) The <a> tag in HTML is used to define hyperlinks, which allow users to navigate to another web page, file, or location within the same page. It is called an "anchor" tag.

Syntax: Link Text

href attribute: The href (hyperlink reference) attribute specifies the destination of the link. It could be a web URL, a local file path, or an anchor within the same page.

Link Text: This is the clickable text or content that the user will see. When clicked, the browser will navigate to the location specified in the href attribute.

The target attribute controls how the linked document will open. The possible values of the target attribute are:

target="_blank": Opens the linked content in a new tab or window. This is commonly used for links that lead to external websites, so users don't lose the current page

target="_self": Opens the link in the same tab or window (this is the default behavior if no target is specified). his is used when you want the link to open in the current window.

target="_parent": Opens the link in the parent frame, useful when your page is embedded inside a frame or iframe.

target="_top": Opens the link in the topmost window or tab, removing any frames if the page is loaded inside frames.

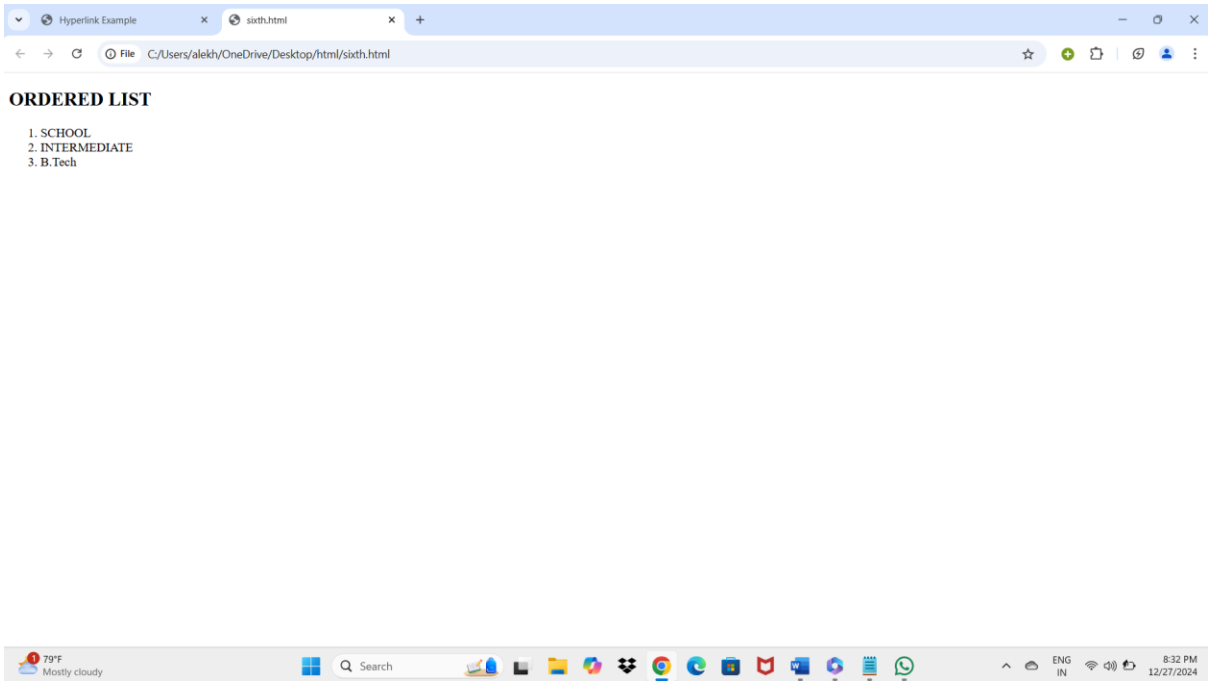
HTML Code:

```
<!DOCTYPE html>
<html>
<head>
   <title>Hyperlink Example</title>
</head>
<body>
   <center><i><b>
      <a href="sixth.html" target="_blank">
         click1!(Open in a new tab)
      </a>
      <br>
      <a href="sixth.html" target="_self">
         click2!(Open in the same tab)
      </a>
      <br>
      <a href="sixth.html" target="_parent">
         click3!(Open in parent frame)
      </a>
      <br>
      <a href="sixth.html" target="_top">
         click4!(Open in topmost window)
      </a>
```

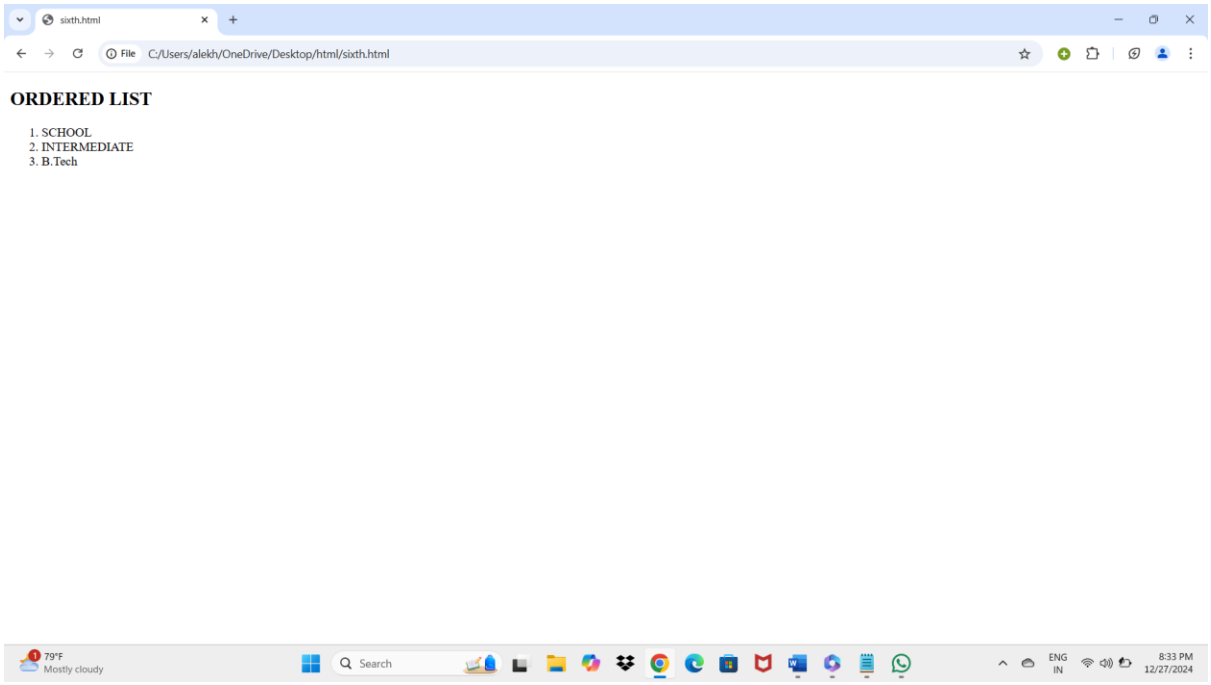
```
        <br>
    </b></i></center>
</body>
</html>
```

Output:

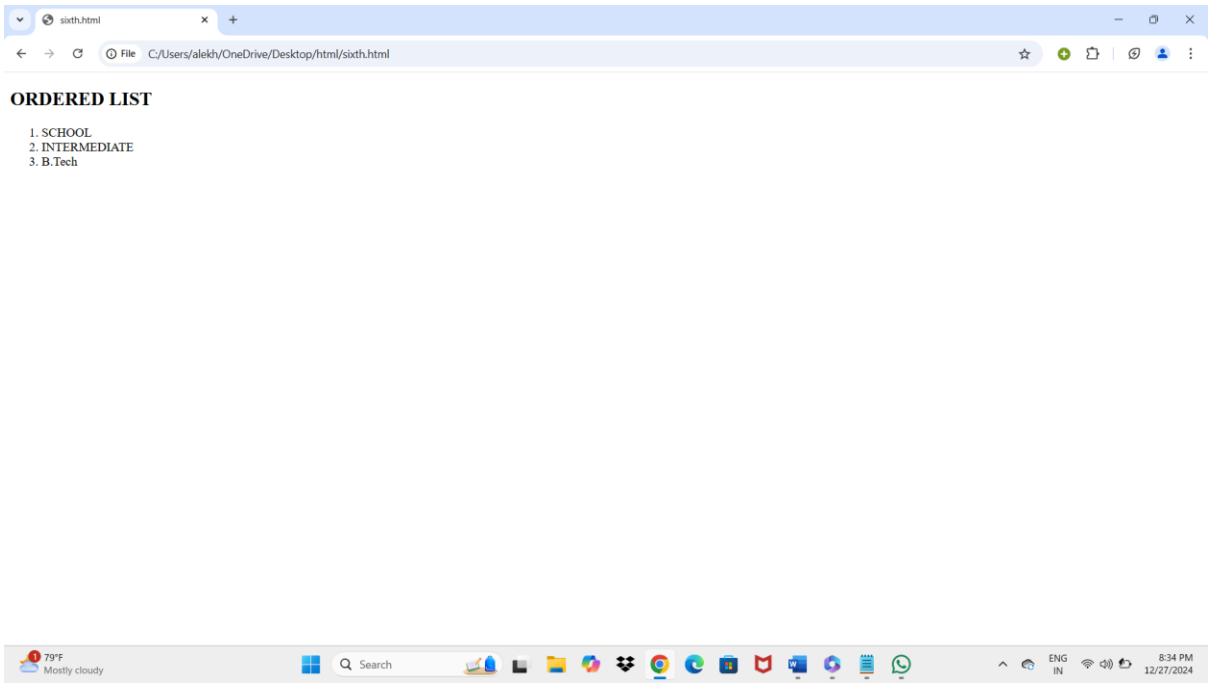
After Clicking click1!(Open in a new tab):



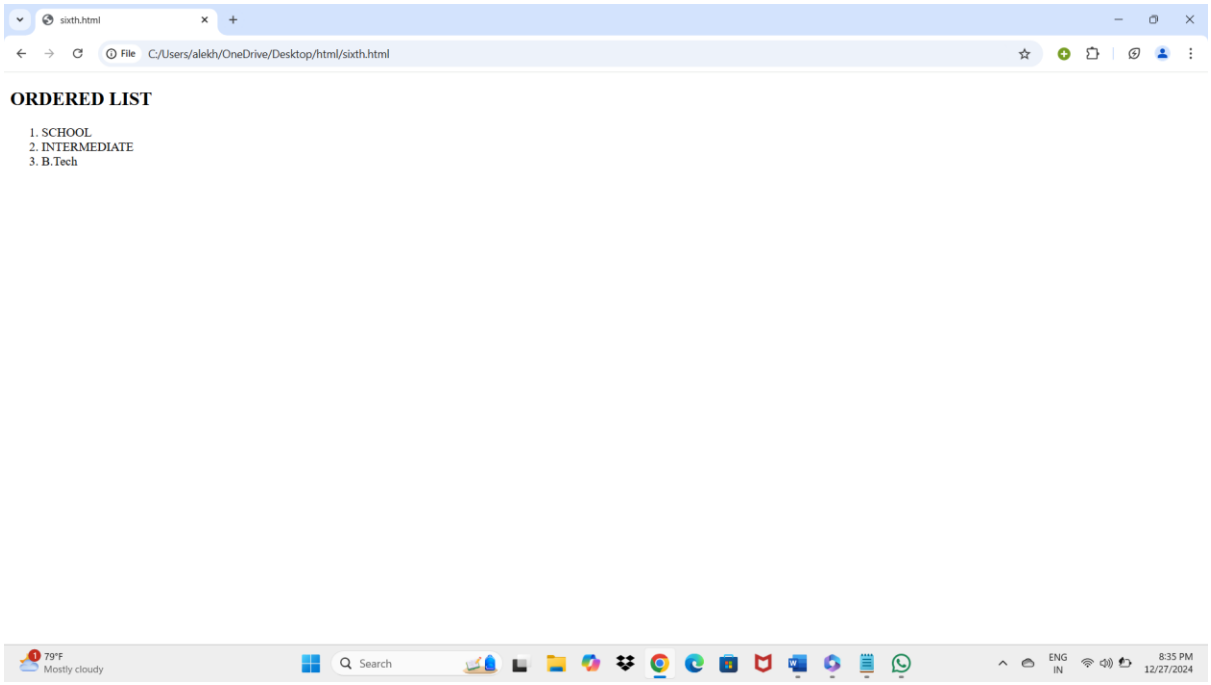
After Clicking click2!(Open in the same tab):



After Clicking click3!(Open in parent frame):



After Clicking click4!(Open in topmost window):

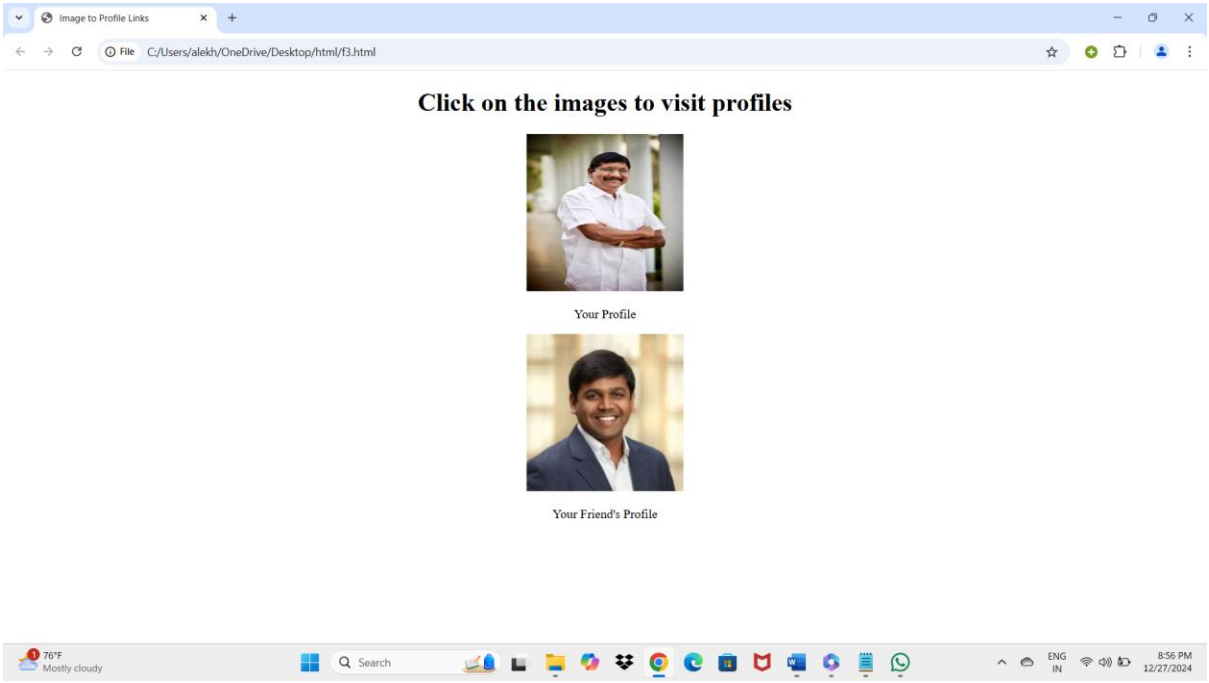

sixth.html
File C:/Users/alekh/OneDrive/Desktop/html/sixth.html
ORDERED LIST
1. SCHOOL
2. INTERMEDIATE
3. B.Tech
79°F
Mostly cloudy
Search
ENG
IN
8:35 PM
12/27/2024

c) Create a HTML document that has your image and your friend's image with a specific height and width. Also when clicked on the images it should navigate to their respective profiles.

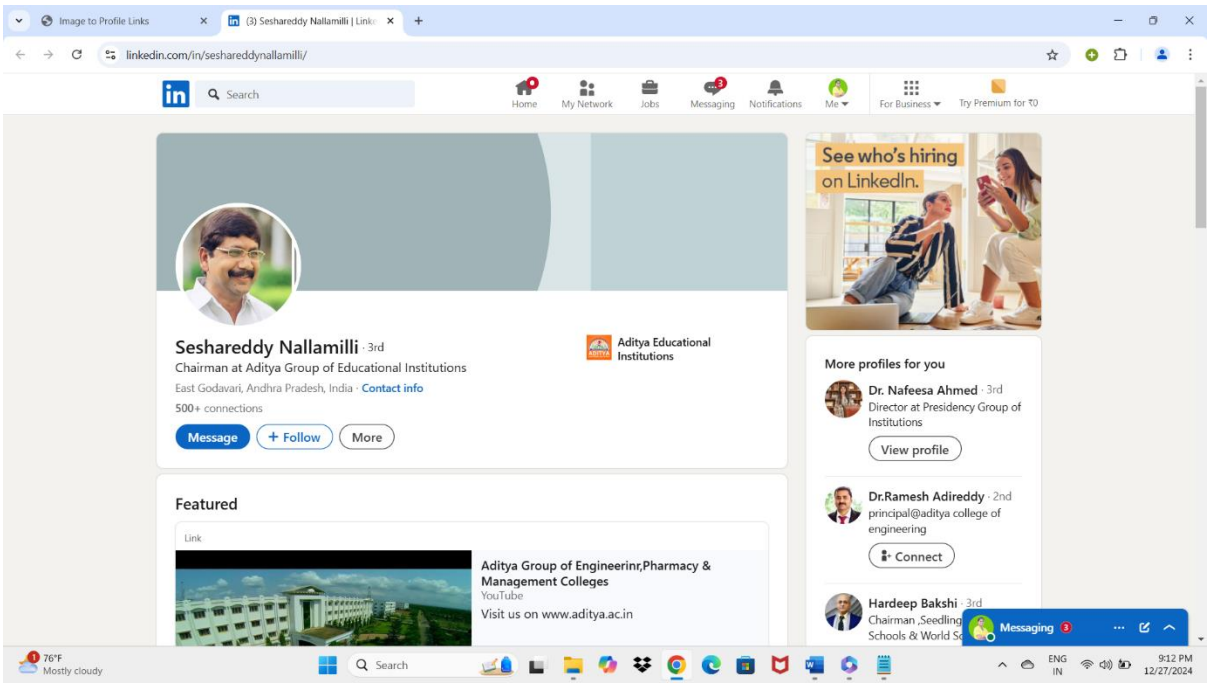
A) HTML Code:

```
<!DOCTYPE html>
<html>
<head>
<title>Image to Profile Links</title>
</head>
<body>
<center>
<h1>Click on the images to visit profiles</h1>
<a href="https://in.linkedin.com/in/seshareddynallamilli" target="_blank">
<img src="chairman.jpg" alt="Your Profile" height="200" width="200">
</a>
<p>Your Profile</p>
<a href="https://in.linkedin.com/in/dr-satish-reddy-nallamilli-116b475a"
target="_blank">
<img src="ViceChairman.jpg" alt="Friend's Profile" height="200"
width="200">
</a>
<p>Your Friend's Profile</p>
</center>
</body>
</html>
```

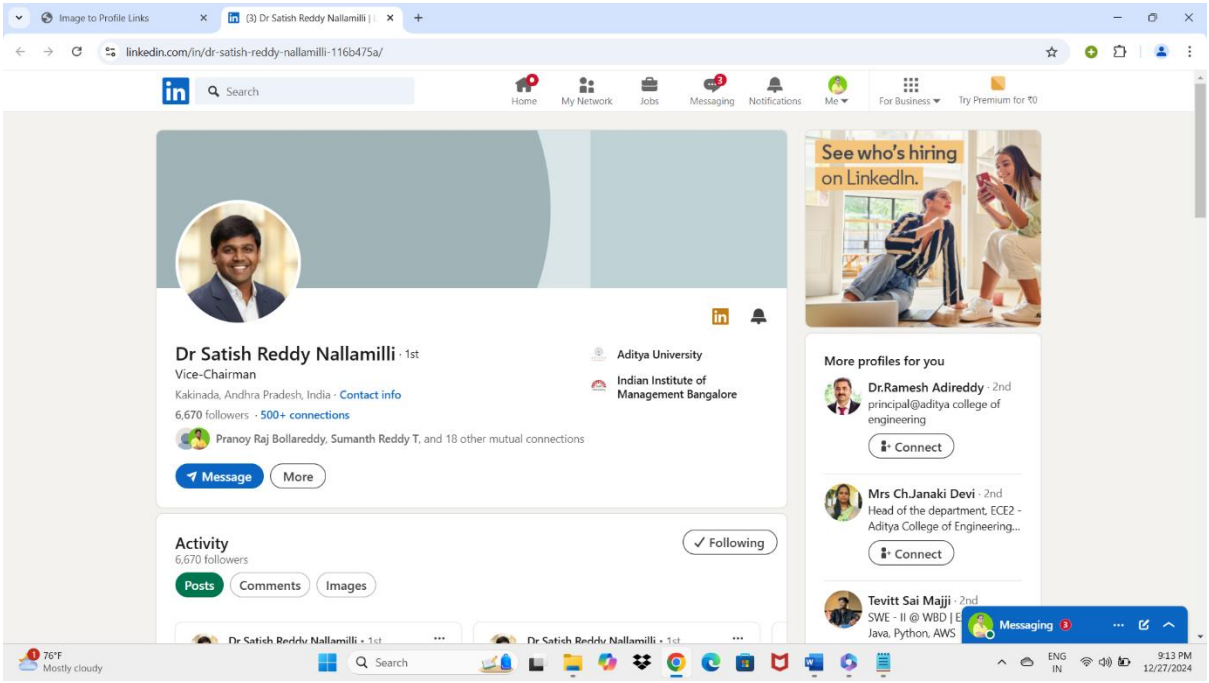
Output:



After Clicking on image above Your Profile:



After Clicking on image above Your Friend's Profile:



d) Write a HTML program, in such a way that, rather than placing large images on a page, the preferred technique is to use thumbnails by setting the height and width parameters to something like to 100*100 pixels. Each thumbnail image is also a link to a full sized version of the image. Create an image gallery using this technique

A) Tables in HTML are used to organize and display data in a structured format.

1. **Tags**:
    a) **<table>:**
        Defines the table itself.
    b) **<tr>:**
        Represents a row within the table.
    c) <**th>:**
        Used to define a header cell, typically bold and centered.
    d) **<td>:**
        Represents a data cell within a row.

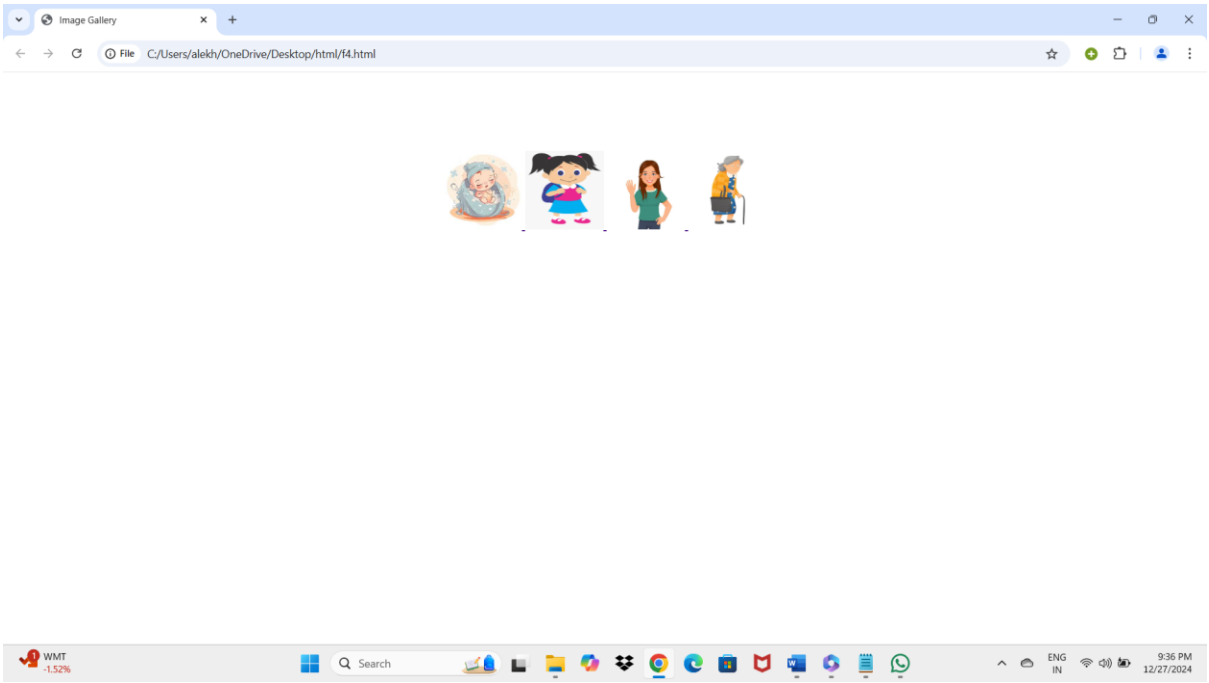
2. **Attributes**:
    a) **border:**
        Specifies the thickness of the table's border.
    b) **rowspan:**
        Merges cells vertically across multiple rows.
    c) **colspan:**
        Merges cells horizontally across multiple columns.

HTML Code:

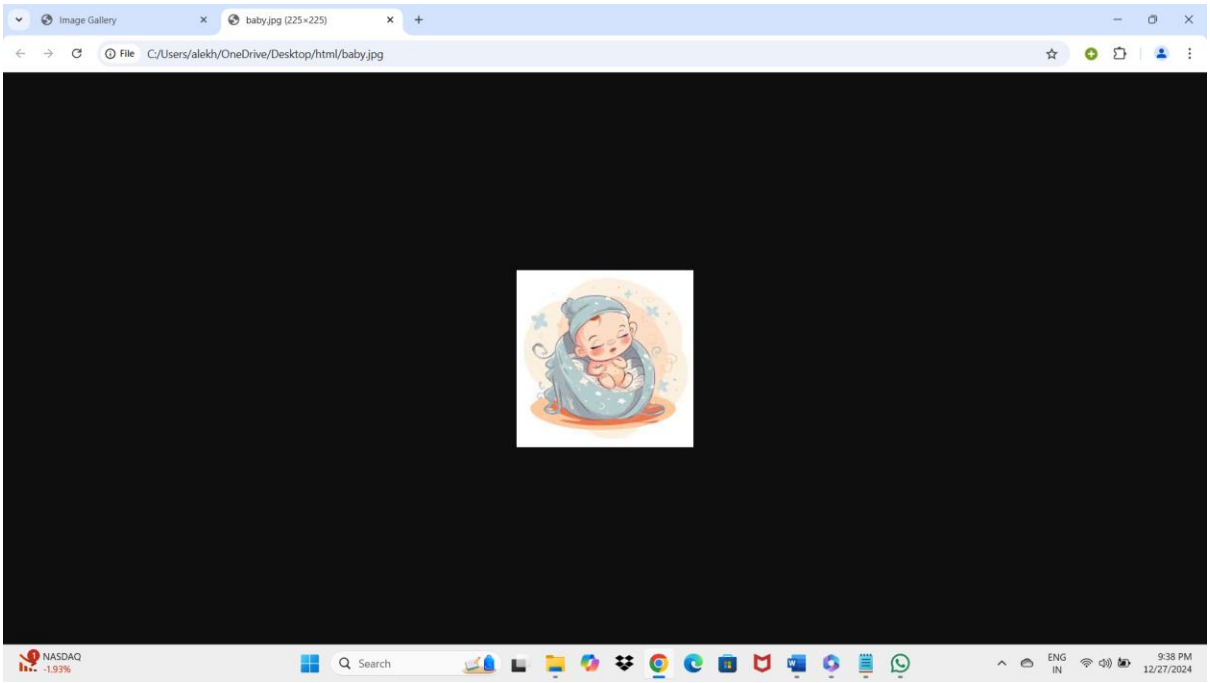
```
<!DOCTYPE html>
<html>
<head>
```

```
    <title>Image Gallery</title>
</head>
<body>
<center><i>
    <b>
    <br><br><br><br><br>
        <a href="baby.jpg" target="_blank">
            <img src="baby.jpg" alt="Image 2" width="100" height="100">
        </a>
        <a href="child.jpg" target="_blank">
            <img src="child.jpg" alt="Image 3" width="100" height="100">
        </a>
        <a href="women.jpg" target="_blank">
            <img src="women.jpg" alt="Image 4" width="100" height="100">
        </a>
        <a href="old.jpg" target="_blank">
            <img src="old.jpg" alt="Image 5" width="100" height="100">
        </a>
    </b>
</i></center>
</body>
</html>
```

Output:



After Clicking Each Image ,We Get Each thumbnail image is linked to a full sized version of the image as follows.



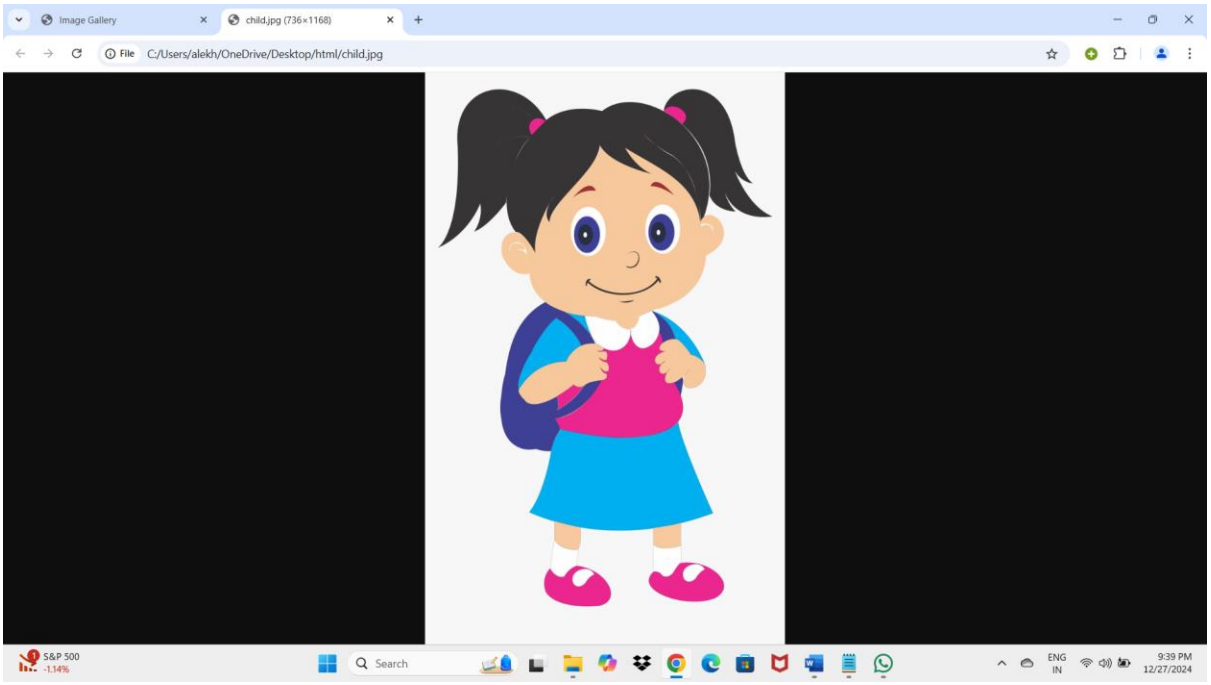


Image Gallery
child.jpg (736×1168)
File
C:/Users/alekh/OneDrive/Desktop/html/child.jpg
S&P 500
-1.14%
Search
ENG
IN
9:39 PM
12/27/2024

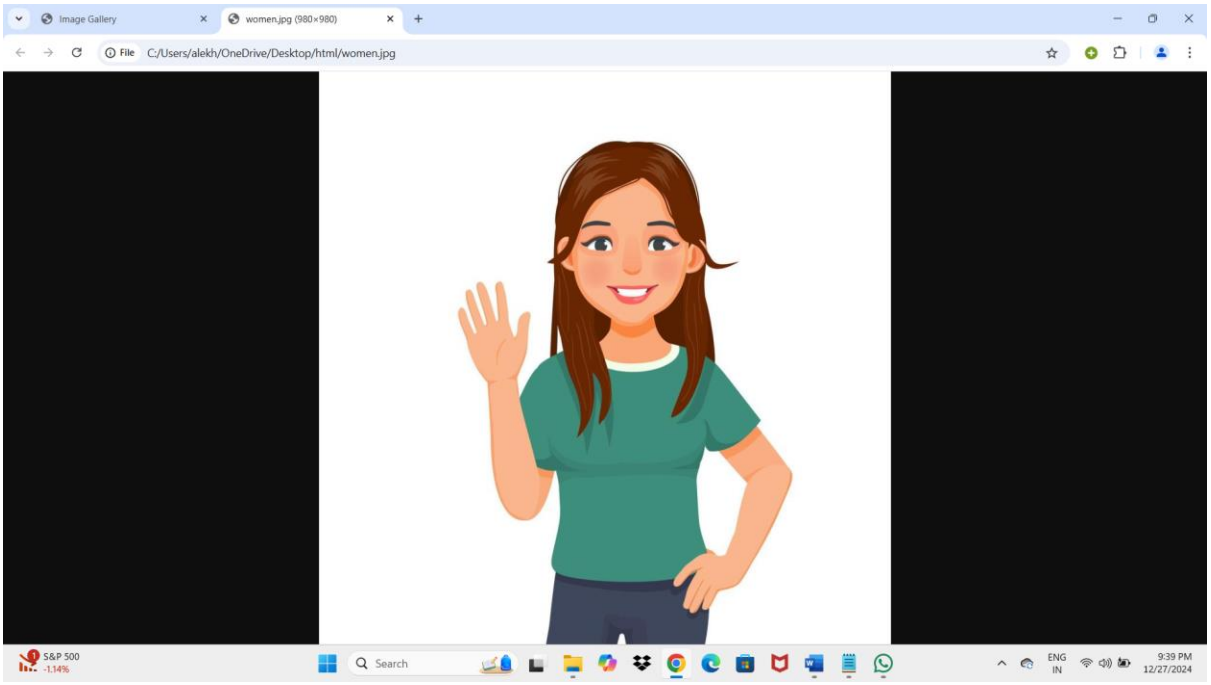
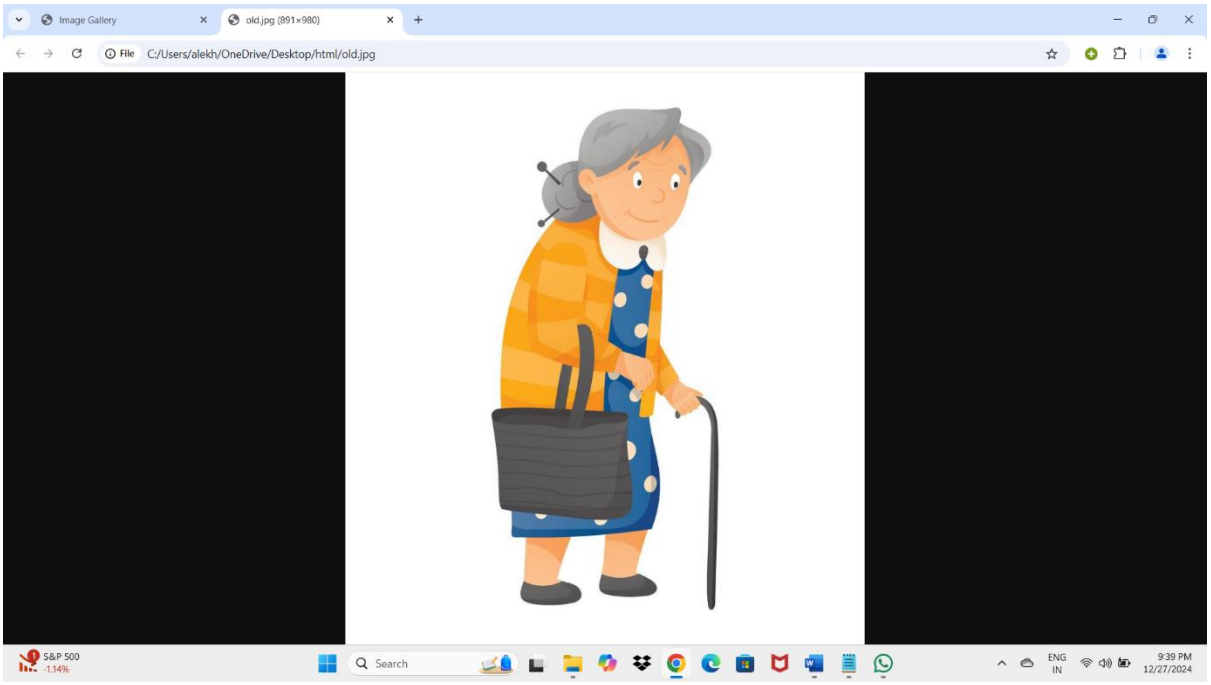


Image Gallery
women.jpg (980×980)
File
C:/Users/alekh/OneDrive/Desktop/html/women.jpg
S&P 500
-1.14%
Search
ENG
IN
9:39 PM
12/27/2024



2. HTML Tables, Forms and Frames

a. Write a HTML program, to explain the working of tables.(use tags:<table> , <tr> , <th> ,<td> , and attributes : border , rowspan , colspan).

A) Description:

1. **<table>:**
    a. This is the main container element for creating a table in an HTML document.
    b. It organizes data into rows and columns.
2. **<tr> (Table Row):**
    a. This element represents a row within the table.
    b. Each <tr> contains table cells, which can either be headers (<th>) or data cells (<td>).
3. **<th> (Table Header):**
    a. Represents a header cell in the table, used for defining titles or headings for columns and rows.
    b. The text inside <th> tags is bold and typically centered by default.
4. **<td> (Table Data):**
    a. Represents a standard data cell in the table.
    b. It contains the actual data or content displayed in the table.
5. **border (Attribute):**
    a. Applied to the <table> tag to create visible lines (borders) around the table, rows, and cells.
    b. You can specify the width of the border using a numeric value (e.g., border="1").
6. **rowspan (Attribute):**
    a. Used to merge (or span) a cell across multiple rows.
    b. This is applied to a <td> or <th> tag and takes a numeric value indicating the number of rows the cell should span.
    c. Useful for creating tables with cells that cover multiple rows.
7. **colspan (Attribute):**
    a. Used to merge (or span) a cell across multiple columns.
    b. This is applied to a <td> or <th> tag and takes a numeric value indicating the number of columns the cell should span.

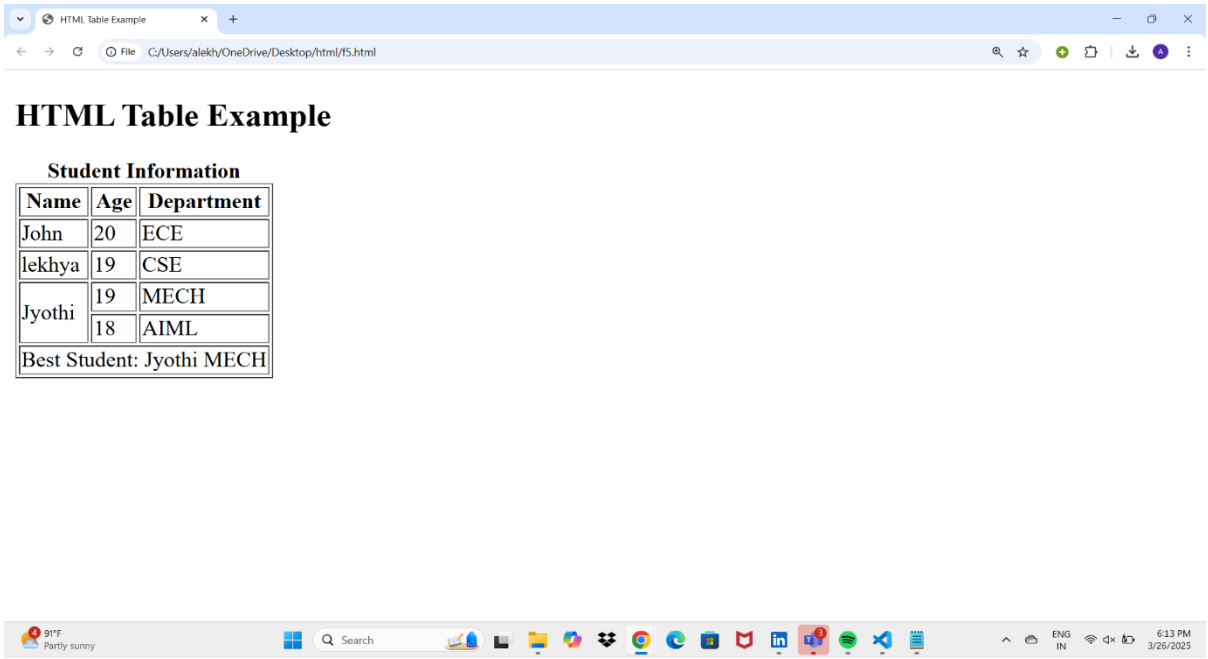
HTML Code:

```
<!DOCTYPE html>

<html>

<head>
```

```
    <title>HTML Table Example</title>
</head>
<body>

    <h2>HTML Table Example</h2>

    <table border="1">
        <caption><strong>Student Information</strong></caption>
        <tr>
            <th>Name</th>
            <th>Age</th>
            <th>Department</th>
        </tr>
        <tr>
            <td>John</td>
            <td>20</td>
            <td>CSE</td>
        </tr>
        <tr>
            <td>lekhya</td>
            <td>19</td>
            <td>CSE</td>
        </tr>
        <tr>
            <td rowspan="2">Jyothi</td>
            <td>19</td>
            <td>MECH</td>
        </tr>
```

```
        <tr>
            <td>18</td>
            <td>AIML</td>
        </tr>
        <tr>
            <td colspan="4">Best Student: Jyothi MECH</td>
        </tr>
    </table>
</body>
</html>
```

OUTPUT:



b. Write a HTML program, to explain the working of tables by preparing a timetable. (Note: Use tag to set the caption to the table & also use cell spacing, cell padding, border, rowspan, colspan etc.)

A) Description:

1. **Caption**: Use the <caption> tag to provide a title or description for the table, such as "Weekly Timetable."

2. **Cellspacing and Cellpadding**: Use the cellspacing attribute to control the space between table cells and the cellpadding attribute to adjust the space between the cell border and its content for better readability.

3. **Border**: Add the border attribute to make the table lines visible and structured.

4. **Rowspan**: Use the rowspan attribute to merge cells vertically, such as combining cells for a session that spans multiple hours.

5. **Colspan**: Use the colspan attribute to merge cells horizontally, like creating a single cell for "Break" across multiple columns.

6. **Headers and Cells**: Include <th> for table headers and <td> for table data to organize the content effectively.

7. **Alignment**: Apply additional CSS or inline styles to customize alignment, colors, or font for enhanced presentation.

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
    <title>Class Timetable</title>
</head>
<body>
```

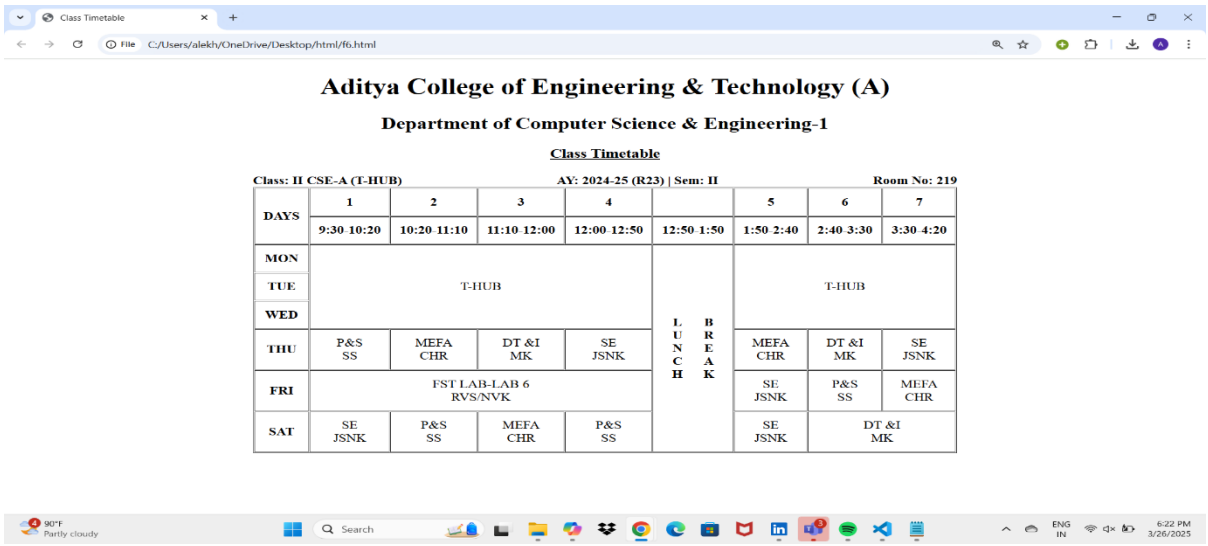
```
<center>
<h1>Aditya College of Engineering & Technology (A)</h1>
<h2>Department of Computer Science & Engineering-1</h2>
<h3><u>Class Timetable</u></h3>
<table border="1" cellspacing="1" cellpadding="10"style="text-align: center;">
    <caption>
        <span style="float: left;"><b>Class: II CSE-A (T-HUB)</b></span>
        <span style="text-align: center; "><b>AY: 2024-25 (R23) | Sem: II</b></span>
        <span style="float: right;"><b>Room No: 219</b></span>
    </caption>
        <tr>
            <th rowspan="2">DAYS</th>
            <th>1</th>
            <th>2</th>
            <th>3</th>
            <th>4</th>
            <th></th>
            <th>5</th>
            <th>6</th>
            <th>7</th>

        </tr>
        <tr>
            <th>9:30-10:20</th>
            <th>10:20-11:10</th>
            <th>11:10-12:00</th>
            <th>12:00-12:50</th>
```

```
            <th>12:50-1:50</th>
            <th>1:50-2:40</th>
            <th>2:40-3:30</th>
            <th>3:30-4:20</th>
        </tr>
        <tr>
            <td><b>MON</b></td>
            <td colspan="4" rowspan="3">T-HUB</td>
            <td rowspan="6"><div style="display: inline-block; padding-right:
20px;">
            <b>
                <div>L</div>
                <div>U</div>
                <div>N</div>
                <div>C</div>
                <div>H</div>
            </b>
            </div>

            <div style="display: inline-block;">
                <b>
                <div>B</div>
                <div>R</div>
                <div>E</div>
                <div>A</div>
                <div>K</div>
            </b>
            </div></td>
```

```
    <td colspan="3" rowspan="3">T-HUB</td>
</tr>
<tr> <td><b>TUE</b></td></tr>
<tr> <td><b>WED</b></td></tr>
<tr>
    <td><b>THU</b></td>
    <td>P&S<br>SS</td>
    <td>MEFA<br>CHR</td>
    <td>DT &I<br>MK</td>
    <td>SE<br>JSNK</td>
    <td>MEFA<br>CHR</td>
    <td>DT &I<br>MK</td>
    <td>SE<br>JSNK</td>
</tr>
<tr>
    <td><b>FRI</b></td>
    <td colspan="4">FST LAB-LAB 6<br>RVS/NVK</td>
    <td>SE<br>JSNK</td>
    <td>P&S<br>SS</td>
    <td>MEFA<br>CHR</td>
</tr>
<tr>
    <td><b>SAT</b></td>
    <td>SE<br>JSNK</td>
    <td>P&S<br>SS</td>
    <td>MEFA<br>CHR</td>
    <td>P&S<br>SS</td>
    <td>SE<br>JSNK</td>
```

```
            <td colspan="2">DT &I<br>MK</td>
        </tr>
    </table>
</center>
</body>
</html>
```

OUTPUT:



Aditya College of Engineering & Technology (A)

Department of Computer Science & Engineering-1

Class Timetable

Class: II CSE-A (T-HUB) **AY: 2024-25 (R23) | Sem: II** **Room No: 219**

DAYS	1	2	3	4		5	6	7
	9:30-10:20	10:20-11:10	11:10-12:00	12:00-12:50	12:50-1:50	1:50-2:40	2:40-3:30	3:30-4:20
MON	T-HUB				LUNCH BREAK	T-HUB		
TUE								
WED								
THU	P&S SS	MEFA CHR	DT &I MK	SE JSNK		MEFA CHR	DT &I MK	SE JSNK
FRI	FST LAB-LAB 6 RVS/NVK					SE JSNK	P&S SS	MEFA CHR
SAT	SE JSNK	P&S SS	MEFA CHR	P&S SS		SE JSNK	DT &I MK	

c. Write a HTML program, to explain the working of forms by designing Registration form. (Note: Include text field, password field, number field, date of birth field, checkboxes, radio buttons, list boxes using <select> &<option> tags,<text area> and two buttons i.e: submit and reset. Use tables to provide a better view).

A) Description:

Using <form> Tag:

The <form> tag is used to create an interactive form where users can input their details.

Elements to Include:

a. **Text Field**: Captures text input like "Name" or "Email".
b. **Password Field**: A text field that masks the input for security.
c. **Number Field**: Allows users to input numeric values like "Age".
d. **Date Field**: Facilitates selection of a date (e.g., Date of Birth).
e. **Checkboxes**: Enables selection of multiple options (e.g., Hobbies).

f. **Radio Buttons**: Allows users to select one option from a group (e.g., Gender).
g. **Drop-down List**: Created with <select> and <option> to display multiple options in a compact form.
h. **Text Area**: Provides a larger text box for input like "Address" or "Comments".
i. **Buttons**:
    i. **Submit Button**: Sends form data for processing.
    ii. **Reset Button**: Clears all fields in the form.

Using <table> for Layout:

A <table> can structure the form fields neatly into rows and columns, enhancing readability.

HTML Code:

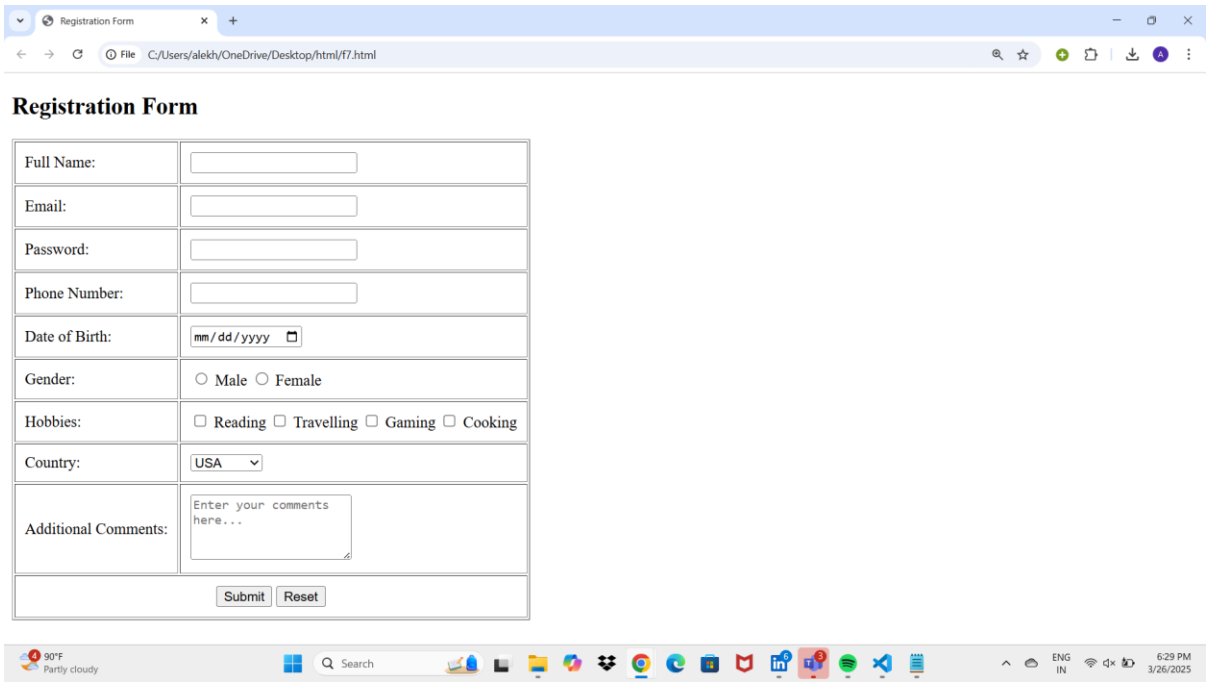
```
<!DOCTYPE html>
<html>
<head>
  <title>Registration Form</title>
</head>
<body>
  <h2>Registration Form</h2>
  <form>
    <table border="1" cellpadding="10">
      <tr>
        <td>Full Name:</td>
        <td><input type="text" id="fullname" name="fullname" required></td>
      </tr>
      <tr>
        <td>Email:</td>
        <td><input type="text" id="email" name="email" required></td>
```

```
        </tr>
        <tr>
            <td>Password:</td>
            <td><input type="password" id="password" name="password" required></td>
        </tr>
        <tr>
            <td>Phone Number:</td>
            <td><input type="number" id="phone" name="phone" required></td>
        </tr>
        <tr>
            <td>Date of Birth:</td>
            <td><input type="date" id="dob" name="dob" required></td>
        </tr>
        <tr>
            <td>Gender:</td>
            <td>
                <input type="radio" id="male" name="gender" value="Male" required> Male
                <input type="radio" id="female" name="gender" value="Female" required> Female
            </td>
        </tr>
        <tr>
            <td>Hobbies:</td>
            <td>
                <input type="checkbox" id="hobby1" name="hobbies" value="Reading"> Reading
                <input type="checkbox" id="hobby2" name="hobbies" value="Travelling"> Travelling
```

```
            <input type="checkbox" id="hobby3" name="hobbies" value="Gaming"> Gaming
            <input type="checkbox" id="hobby4" name="hobbies" value="Cooking"> Cooking
        </td>
    </tr>
    <tr>
        <td>Country:</td>
        <td>
            <select id="country" name="country" required>
                <option value="USA">USA</option>
                <option value="Canada">Canada</option>
                <option value="India">India</option>
                <option value="Australia">Australia</option>
                <option value="UK">UK</option>
            </select>
        </td>
    </tr>
    <tr>
        <td>Additional Comments:</td>
        <td><textarea id="comments" name="comments" rows="4" placeholder="Enter your comments here..."></textarea></td>
    </tr>
    <tr>
        <td colspan="2" style="text-align: center;">
            <input type="submit" value="Submit">
            <input type="reset" value="Reset">
        </td>
    </tr>
```

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

OUTPUT:



d. Write a HTML program, to explain the working of frames, such that page is to be divided into 3 parts on either direction. (Note: first frame -> image , second frame -> paragraph, third frame -> hyperlink. And also make sure of using "no frame" attribute such that frames to be fixed)

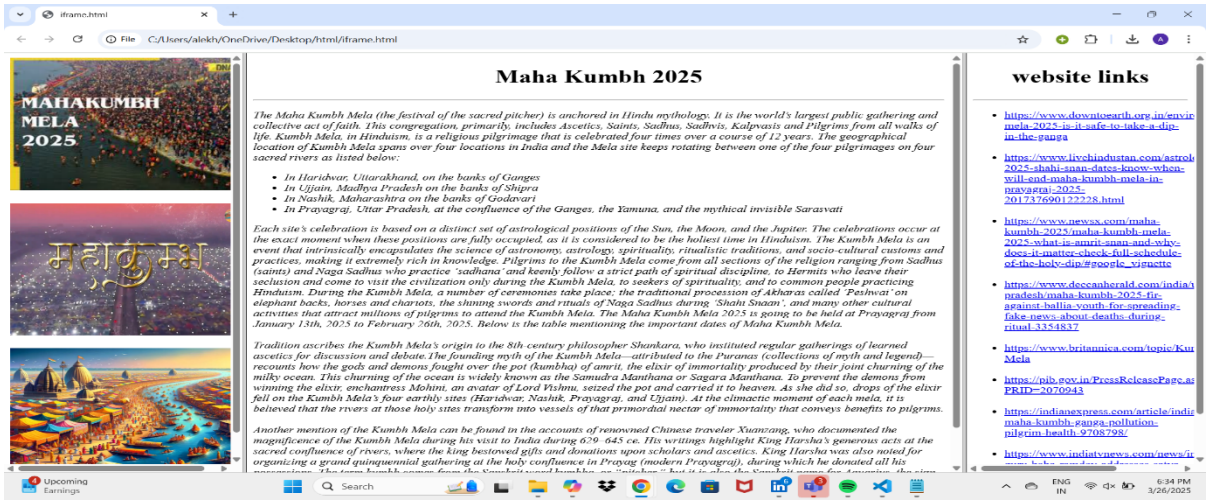
A) Description:

Frames in HTML enable the division of a webpage into sections for displaying independent content. The element organizes the page layout, and specifies each section's content.

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
   <frameset cols="20%,60%,*">
      <frame name="top" src="img.html" style="align-content: center;"/>
      <frame name="main" src="para.html"/>
      <frame name="bottom" src="links.html"/>
   </frameset>
</head>
<body>


</body>
</html>
```

OUTPUT:



The Maha Kumbh Mela (the festival of the sacred pitcher) is anchored in Hindu mythology. It is the world's largest public gathering and collective act of faith. This congregation, primarily, includes Ascetics, Saints, Sadhus, Sadhvis, Kalpvasis and Pilgrims from all walks of life. Kumbh Mela, in Hinduism, is a religious pilgrimage that is celebrated four times over a course of 12 years. The geographical location of Kumbh Mela spans over four locations in India and the Mela site keeps rotating between one of the four pilgrimages on four sacred rivers as listed below:

- *In Haridwar, Uttarakhand, on the banks of Ganges*
- *In Ujjain, Madhya Pradesh on the banks of Shipra*
- *In Nashik, Maharashtra on the banks of Godavari*
- *In Prayagraj, Uttar Pradesh, at the confluence of the Ganges, the Yamuna, and the mythical invisible Saraswati*

Each site's celebration is based on a distinct set of astrological positions of the Sun, the Moon, and the Jupiter. The celebrations occur at the exact moment when these positions are fully occupied, as it is considered to be the holiest time in Hinduism. The Kumbh Mela is an event that intrinsically encapsulates the science of astronomy, astrology, spirituality, ritualistic traditions, and socio-cultural customs and practices, making it extremely rich in knowledge. Pilgrims to the Kumbh Mela come from all sections of the religion ranging from Sadhus (saints) and Naga Sadhus who practice 'sadhana' and keenly follow a strict path of spiritual discipline, to Hermits who leave their seclusion and come to visit the civilization only during the Kumbh Mela, to seekers of spirituality, and to common people practicing Hinduism. During the Kumbh Mela, a number of ceremonies take place; the traditional procession of Akharas called 'Peshwai' on elephant backs, horses and chariots, the shining swords and rituals of Naga Sadhus during 'Shahi Snaan', and many other cultural activities that attract millions of pilgrims to attend the Kumbh Mela. The Maha Kumbh Mela 2025 is going to be held at Prayagraj from January 13th, 2025 to February 26th, 2025. Below is the table mentioning the important dates of Maha Kumbh Mela.

Tradition ascribes the Kumbh Mela's origin to the 8th-century philosopher Shankara, who instituted regular gatherings of learned ascetics for discussion and debate.The founding myth of the Kumbh Mela—attributed to the Puranas (collections of myth and legend)—recounts how the gods and demons fought over the pot (kumbha) of amrit, the elixir of immortality produced by their joint churning of the milky ocean. This churning of the ocean is widely known as the Samudra Manthana or Sagara Manthana. To prevent the demons from winning the elixir, enchantress Mohini, an avatar of Lord Vishnu, seized the pot and carried it to heaven. As she did so, drops of the elixir fell on the Kumbh Mela's four earthly sites (Haridwar, Nashik, Prayagraj, and Ujjain). At the climactic moment of each mela, it is believed that the rivers at those holy sites transform into vessels of that primordial nectar of immortality that conveys benefits to pilgrims.

Another mention of the Kumbh Mela can be found in the accounts of renowned Chinese traveler Xuanzang, who documented the magnificence of the Kumbh Mela during his visit to India during 629–645 ce. His writings highlight King Harsha's generous acts at the sacred confluence of rivers, where the king bestowed gifts and donations upon scholars and ascetics. King Harsha was also noted for organizing a grand quinquennial gathering at the holy confluence in Prayag (modern Prayagraj), during which he donated all his

- https://www.downtoearth.org.in/environment/maha-kumbh-mela-2025-is-it-safe-to-take-a-dip-in-the-ganga
- https://www.livehindustan.com/astrology/story-maha-kumbh-mela-2025-shahi-snan-dates-know-when-will-end-maha-kumbh-mela-in-prayagraj-2025-20173769012228.html
- https://www.newsx.com/maha-kumbh-2025/maha-kumbh-mela-2025-what-is-amrit-snan-and-why-does-it-matter-check-full-schedule-of-the-holy-dip/#google_vignette
- https://www.deccanherald.com/india/uttar-pradesh/maha-kumbh-2025-fir-against-ballia-youth-for-spreading-fake-news-about-deaths-during-ritual-3354837
- https://www.britannica.com/topic/Kumbh-Mela
- https://pib.gov.in/PressReleasePage.aspx?PRID=2070943
- https://indianexpress.com/article/india/maha-kumbh-ganga-pollution-pilgrim-health-9708798/
- https://www.indiatvnews.com/news/india/maha-kumbh-mandir-addresses-setup

3. HTML 5 and Cascading Style Sheets, Types of CSS

a. Write a HTML program, that makes use <article>, <aside>, <figure>, <figcaption>, <footer>, <header>, <main>, <nav>, <section>, , tags.

A) Description:

1. **<article>**: Defines independent, self-contained content, such as a blog post or news article. It can be distributed and reused independently.
2. **<aside>**: Represents content tangentially related to the main content, such as sidebars, callouts, or related links.
3. **<figure>**: Encapsulates content like images, illustrations, diagrams, or charts. It is often used with a <figcaption>.
4. **<figcaption>**: Provides a caption or description for the content inside a <figure> tag, improving context.
5. **<footer>**: Represents the footer of a section or page, typically containing metadata, copyright info, or navigation links.
6. **<header>**: Represents introductory content, such as a page heading or a title section, often including navigation links.
7. **<main>**: Represents the main content of a page, excluding repeated content like sidebars or headers.
8. **<nav>**: Defines a set of navigation links, often for menus or tables of contents.
9. **<section>**: Groups related content within a document, with a specific purpose or theme.
10. ** **: A non-semantic container used for grouping content, often styled with CSS.
11. ****: A non-semantic inline container used for styling or applying scripts to small pieces of text.

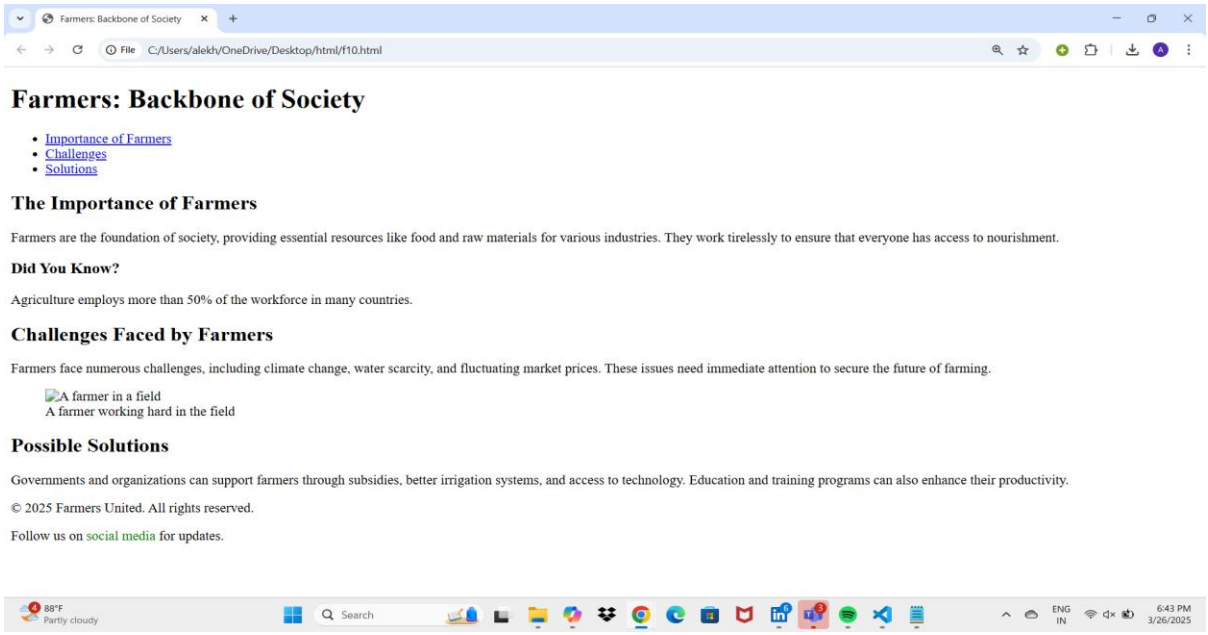
HTML Code:

```
<!DOCTYPE html>
<html>
<head>
    <title>Farmers: Backbone of Society</title>
</head>
<body>
    <header>
        <h1>Farmers: Backbone of Society</h1>
        <nav>
            <ul>
                <li><a href="#importance">Importance of Farmers</a></li>
                <li><a href="#challenges">Challenges</a></li>
                <li><a href="#solutions">Solutions</a></li>
            </ul>
        </nav>
    </header>

    <main>
        <section id="importance">
            <h2>The Importance of Farmers</h2>
            <article>
                <p>Farmers are the foundation of society, providing essential resources like food and raw materials for various industries. They work tirelessly to ensure that everyone has access to nourishment.</p>
            </article>
        </section>

        <aside>
            <h3>Did You Know?</h3>
```

```
        <p>Agriculture employs more than 50% of the workforce in many countries.</p>
    </aside>

    <section id="challenges">
        <h2>Challenges Faced by Farmers</h2>
        <article>
            <p>Farmers face numerous challenges, including climate change, water scarcity, and fluctuating market prices. These issues need immediate attention to secure the future of farming.</p>
        </article>
    </section>

    <figure>
        <img src="farmer.jpg" alt="A farmer in a field">
        <figcaption>A farmer working hard in the field</figcaption>
    </figure>

    <section id="solutions">
        <h2>Possible Solutions</h2>
        <article>
            <p>
                Governments and organizations can support farmers through subsidies, better irrigation systems, and access to technology. Education and training programs can also enhance their productivity.
            </p>
        </article>
    </section>
</main>
```

```
    <footer>
        <p>&copy; 2025 Farmers United. All rights reserved.</p>
        <div>
            <p>Follow us on <span style="color: green;">social media</span> for updates.</p>
        </div>
    </footer>
</body>
</html>
```

OUTPUT:



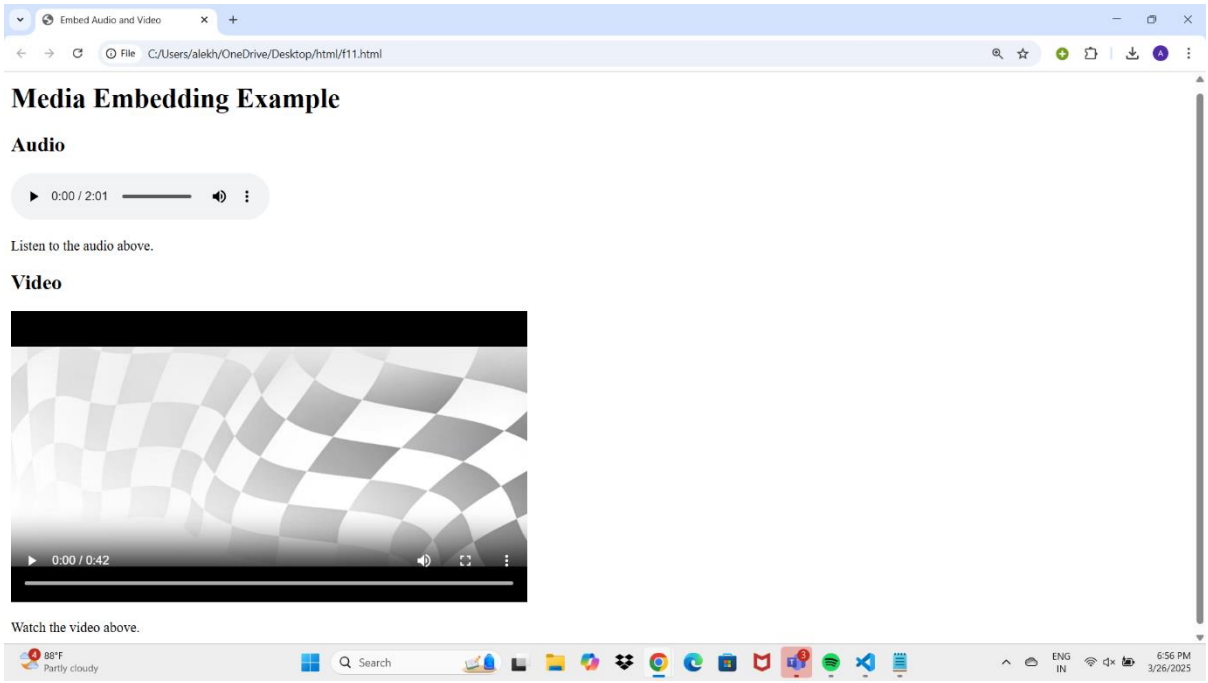
b. Write a HTML program, to embed audio and video into HTML web page.

A) HTML Code:

```
<!DOCTYPE html>
<html>
<head>
    <title>Embed Audio and Video</title>
</head>
<body>
    <h1>Media Embedding Example</h1>
```

```
    <section>
        <h2>Audio</h2>
        <audio controls>
            <source src="audio-file.mp3" type="audio/mpeg">
            Your browser does not support the audio element.
        </audio>
        <p>Listen to the audio above.</p>
    </section>
    <section>
        <h2>Video</h2>
        <video controls width="600">
            <source src="video-file.mp4" type="video/mp4">
            Your browser does not support the video element.
        </video>
        <p>Watch the video above.</p>
    </section>

</body>
</html>
```

OUTPUT:



c. Write a program to apply different types (or levels of styles or style specification formats) - inline, internal, external styles to HTML elements. (identify selector, property and value).

A) **Description:**

1. **Inline Styles**:

    a. These are defined within the HTML tag using the style attribute.
    b. Useful for quick, one-off style changes to a single element.
    c. Example properties include color, font-size, text-align.
2. **Internal Styles**:
    a. Written inside a <style> tag, typically within the <head> section of the HTML file.
    b. Styles are scoped to the specific document and allow grouping of styles for multiple elements.
    c. Use selectors like .class for class-based styling or #id for unique element styling.
3. **External Styles**:
    a. Defined in a separate CSS file with a .css extension and linked to the HTML document via the <link> tag.
    b. Offers the advantage of centralized style management for multiple web pages.
    c. Enables reusability of styles across the entire website, ensuring consistency.

Key Concepts:

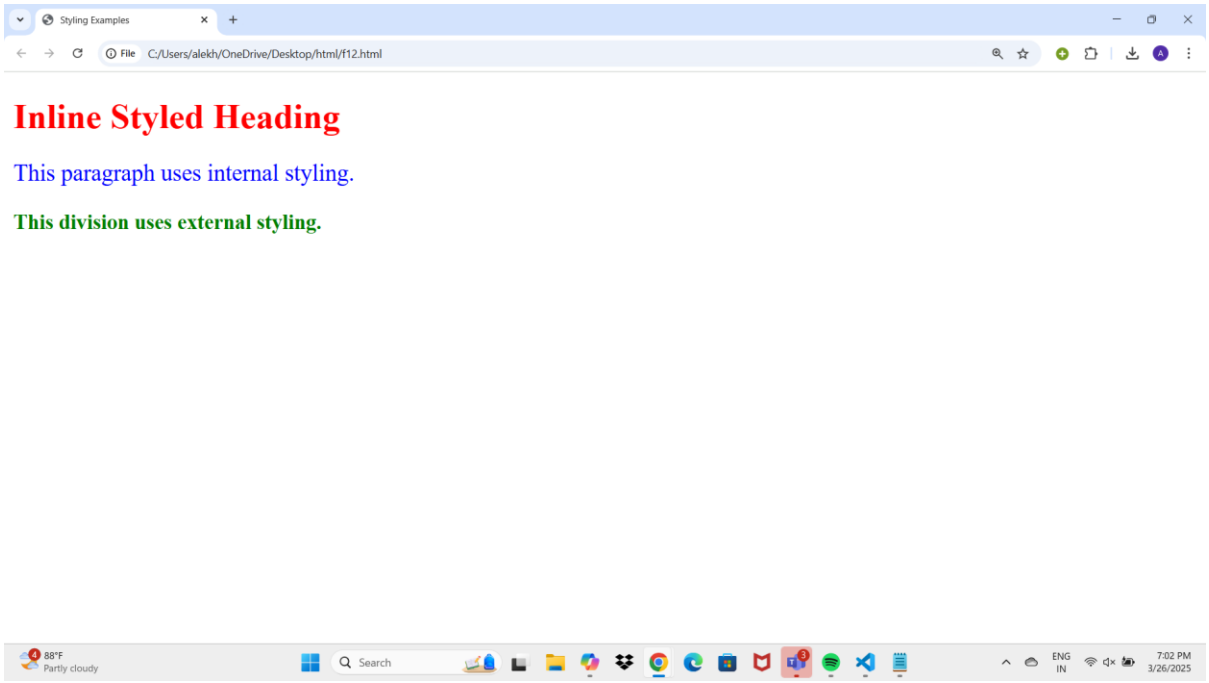
- **Selector**: Targets specific HTML elements (e.g., .classname , #idname, tagname).
- **Property**: Determines the type of style (e.g., background-color, border, margin).
- **Value**: Specifies the actual styling (e.g., blue, solid 1px, 10px).

HTML Code:

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

```
    <title>Styling Examples</title>
    <link rel="stylesheet" href="styles.css">
    <style>
        .internal-style {
            color: blue;
            font-size: 20px;
        }
    </style>
</head>
<body>
    <h1 style="color: red; font-size: 30px;">Inline Styled Heading</h1>
    <p class="internal-style">This paragraph uses internal styling.</p>
    <div class="external-style">This division uses external styling.</div>
</body>
</html>
```

OUTPUT:



4. Selector forms

a. Write a program to apply different types of selector forms

i. Simple selector (element, id, class, group, universal) ii. Combinator selector (descendant, child, adjacent sibling, general sibling) iii. Pseudo-class selector iv. Pseudo-element selector v. Attribute selector

A) Description:

1. **Simple Selector**: Targets elements directly by name, id, class, or groups of elements. Types:
   a) **Element Selector**: Targets specific HTML tags (e.g., p, h1).
   b) **ID Selector**: Targets elements by their unique id attribute (#idname).
   c) **Class Selector**: Targets elements by class (.classname).
   d) **Group Selector**: Targets multiple elements (e.g., p, h1).
   e) **Universal Selector**: Targets all elements (*).
2. **Combinator Selector**: Defines relationships between elements based on structure:
   a) **Descendant Selector**: Targets all descendants within a container
   b) **Child Selector**: Targets direct children only (e.g., div > p).
   c) **Adjacent Sibling Selector**: Targets the immediate sibling
   d) **General Sibling Selector**: Targets all siblings (e.g., h1 ~ p).
3. **Pseudo-Class Selector**:
   a) Targets elements based on specific states or conditions.
   b) Examples: :hover, :focus, :nth-child(), :last-of-type.
4. **Pseudo-Element Selector**:
   a) Targets specific parts of an element.
   b) Examples: ::before, ::after, ::first-letter, ::selection.
5. **Attribute Selector**:
   a) Targets elements based on attributes and their values.
   b) Types:
      i. [attribute]: Selects elements with a specified attribute.
      ii. [attribute=value]: Selects elements with a specific attribute value.
      iii. [attribute^=value], [attribute$=value], [attribute*=value]: Targets partial matches of attribute values (starts with, ends with, contains).

HTML Code:

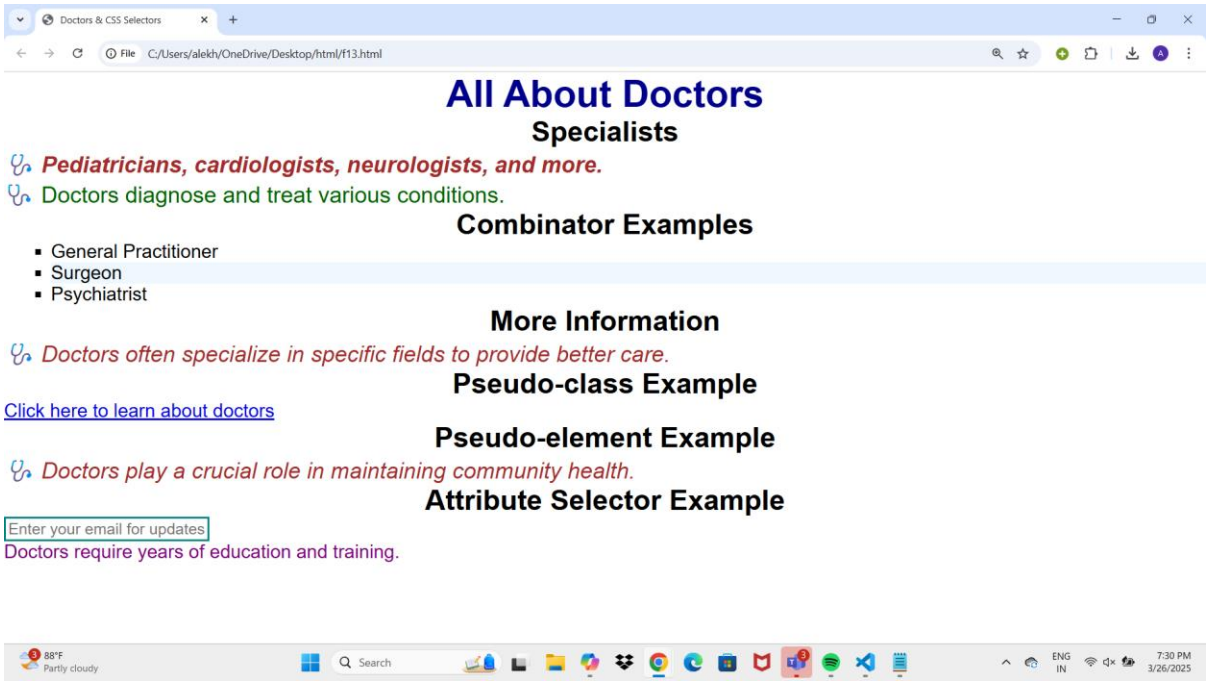
```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Doctors & CSS Selectors</title>
<style>
  body {
    font-family: Arial, sans-serif;
  }
  #doctor-heading {
    color: darkblue;
  }
  .specialist {
    font-weight: bold;
  }
  h1, h2 {
    text-align: center;
  }
  * {
    margin: 0;
  }
  div p {
    color: darkgreen;
  }
  ul > li {
    list-style: square;
  }
  h2 + p {
    color: brown;
  }
  h2 ~ p {
    font-style: italic;
  }
```

```
    a:hover {
      color: crimson;
    }

    li:nth-child(even) {
      background-color: #f0f8ff;
    }
    p::first-line {
      font-size: 1.2em;
    }
    p::before {
      content: '🩺 ';
    }
    [type="email"] {
      border: 2px solid teal;
    }
    [data-role="doctor-info"] {
      color: purple;     }
  </style>
</head>
<body>
  <h1 id="doctor-heading">All About Doctors</h1>

  <h2>Specialists</h2>
  <p class="specialist">Pediatricians, cardiologists, neurologists, and more.</p>
  <div>
    <p>Doctors diagnose and treat various conditions.</p>
  </div>
  <h2>Combinator Examples</h2>
  <ul>
```

```
    <li>General Practitioner</li>
    <li>Surgeon</li>
    <li>Psychiatrist</li>
  </ul>
  <h2>More Information</h2>
  <p>Doctors often specialize in specific fields to provide better care.</p>
  <h2>Pseudo-class Example</h2>
  <a href="#">Click here to learn about doctors</a>
  <h2>Pseudo-element Example</h2>
  <p>Doctors play a crucial role in maintaining community health.</p>
  <h2>Attribute Selector Example</h2>
  <input type="email" placeholder="Enter your email for updates">
  <div data-role="doctor-info">Doctors require years of education and training.</div>
</body>
</html>
```

OUTPUT:



5. CSS with Color, Background, Font, Text and CSS Box Model

a. Write a program to demonstrate the various ways you can reference a color in CSS

A) Description:

1. **Named Colors**: Predefined names like aqua, black, coral, etc., e.g., color: blue;.

2. **Hexadecimal (HEX)**: Represents colors in a combination of red, green, and blue components as hexadecimal values, e.g., #FFFFFF (white) or #000000 (black). Shorthand like #FFF is also supported.

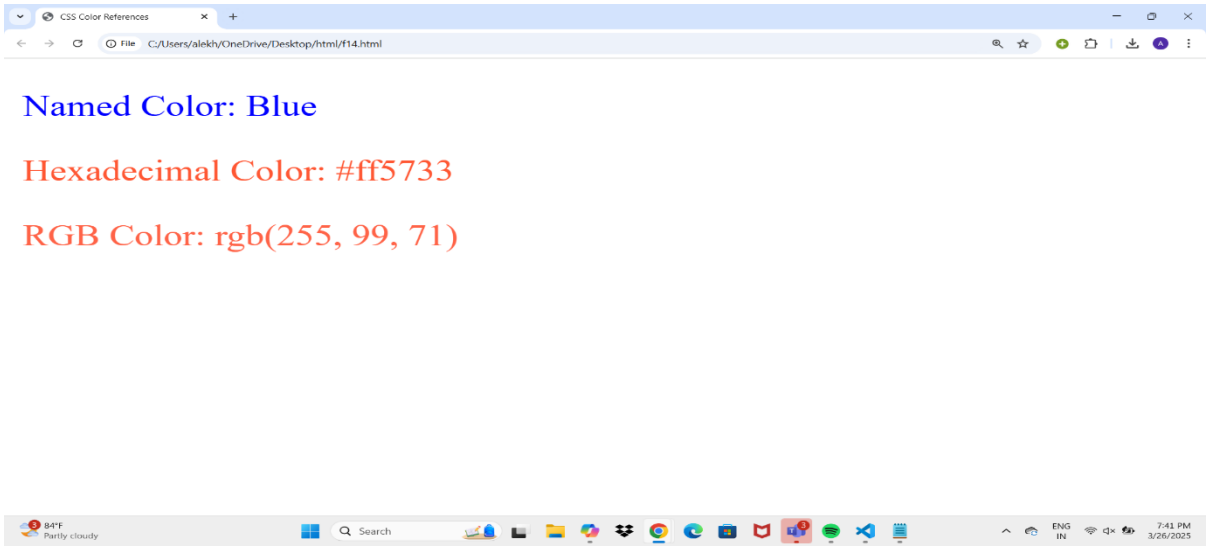
3. **RGB**: Specifies colors using rgb() with values for red, green, and blue ranging from 0 to 255, e.g., rgb(255, 0, 0) for red.

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Color References</title>
  <style>
    .named-color {
      color: blue;
    }
    .hex-color {
      color: #ff5733;
    }
    .rgb-color {
      color: rgb(255, 99, 71);
```

```
    }
  </style>
</head>
<body>
  <p class="named-color">Named Color: Blue</p>
  <p class="hex-color">Hexadecimal Color: #ff5733</p>
  <p class="rgb-color">RGB Color: rgb(255, 99, 71)</p>
</body>
</html>
```

OUTPUT:



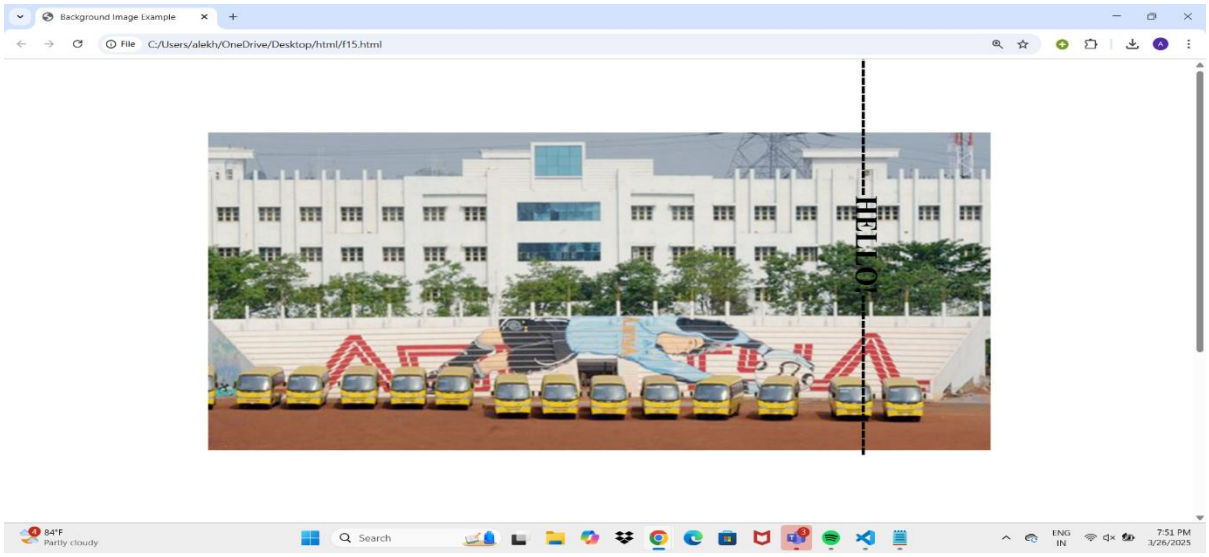
b. Write a CSS rule that places a background image halfway down the page, tilting it horizontally. The image should remain in place when the user scrolls up or down

A) HTML Code:

```
<!DOCTYPE html>
<html>
<head>
```

```
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Background Image Example</title>
  <style>
    body {
      background-image: url('img.jpg');
      background-position: center 50%;
      background-repeat: no-repeat;
      background-attachment: fixed;
      transform: rotate(90deg);
      height: 100vh;
    }
  </style>
</head>
<body>
  <center></center>
  <h1>-----------------------------------------HELLO!-------------------</h1>
  </center>
</body>
</html>
```

OUTPUT:



c. Write a program using the following terms related to CSS font and text: i. font-size ii. font-weight iii. font-style iv. text-decoration v. text-transformation vi. text-alignment

A) Description:

1. **font-size**: Determines the size of the text. You can use units like pixels (px), em, rem, or percentages. Example: font-size: 20px;.

2. **font-weight**: Controls the boldness of the text. Values include normal, bold, or numbers like 100 (light) to 900 (extra bold). Example: font-weight: bold;.

3. **font-style**: Defines the style of the font. Common values include normal, italic, and oblique. Example: font-style: italic;.

4. **text-decoration**: Adds decorative lines to the text. Options include underline, line-through, overline, or none. Example: text-decoration: underline;.

5. **text-transform**: Specifies the capitalization of the text. Values include uppercase (all caps), lowercase (all small letters), and capitalize (first letter of each word capitalized). Example: text-transform: uppercase;.

6. **text-align**: Aligns text horizontally within its container. Common values include left, right, center, and justify. Example: text-align: center;.

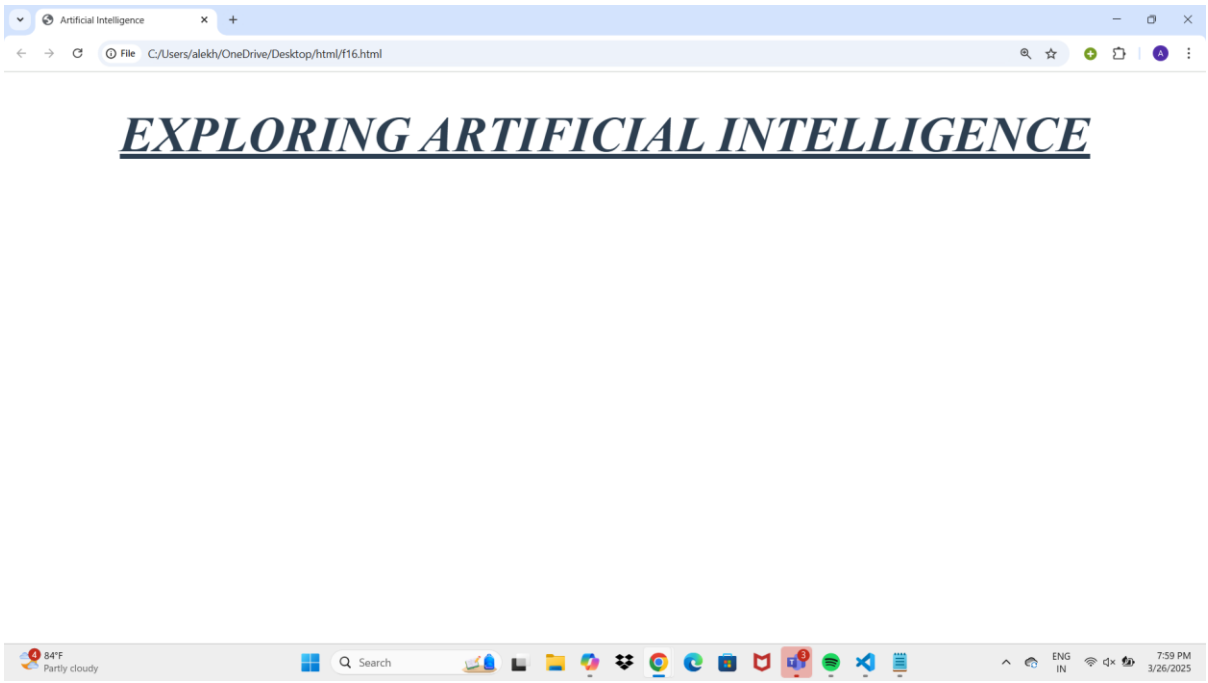
HTML Code:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Artificial Intelligence</title>
    <style>
        .ai-heading {
            font-size: 32px;
            font-weight: bold;
            font-style: italic;
            text-decoration: underline;
            text-transform: uppercase;
            text-align: center;
            color: #2c3e50;
        }
    </style>
</head>
<body>
    <h1 class="ai-heading">Exploring Artificial Intelligence</h1>
```

</body>

</html>

OUTPUT:



d. Write a program, to explain the importance of CSS Box model using i. Content ii. Border iii. Margin iv. Padding

A) Description:

The CSS Box Model is essential for designing and formatting elements on a webpage. It consists of:

1. **Content**:
    a. The actual text, image, or element inside the box.
    b. It is the core part of the box where the information is displayed.
2. **Padding**:
    a. The space between the content and the border.
    b. It creates breathing room around the content, ensuring the text or elements do not touch the border.
    c. Example: padding: 20px; adds 20 pixels of space inside the border.
3. **Border**:
    a. A line or edge that surrounds the content and padding.

   b. It provides a visible boundary for the element, which can be styled (e.g., solid, dotted, dashed).
   c. Example: border: 5px solid #3498db; adds a 5-pixel blue solid border.
4. **Margin**:
   a. The space outside the border that separates the element from other elements on the page.
   b. It creates layout spacing and avoids overlap between different elements.
   c. Example: margin: 30px; adds 30 pixels of space outside the border.

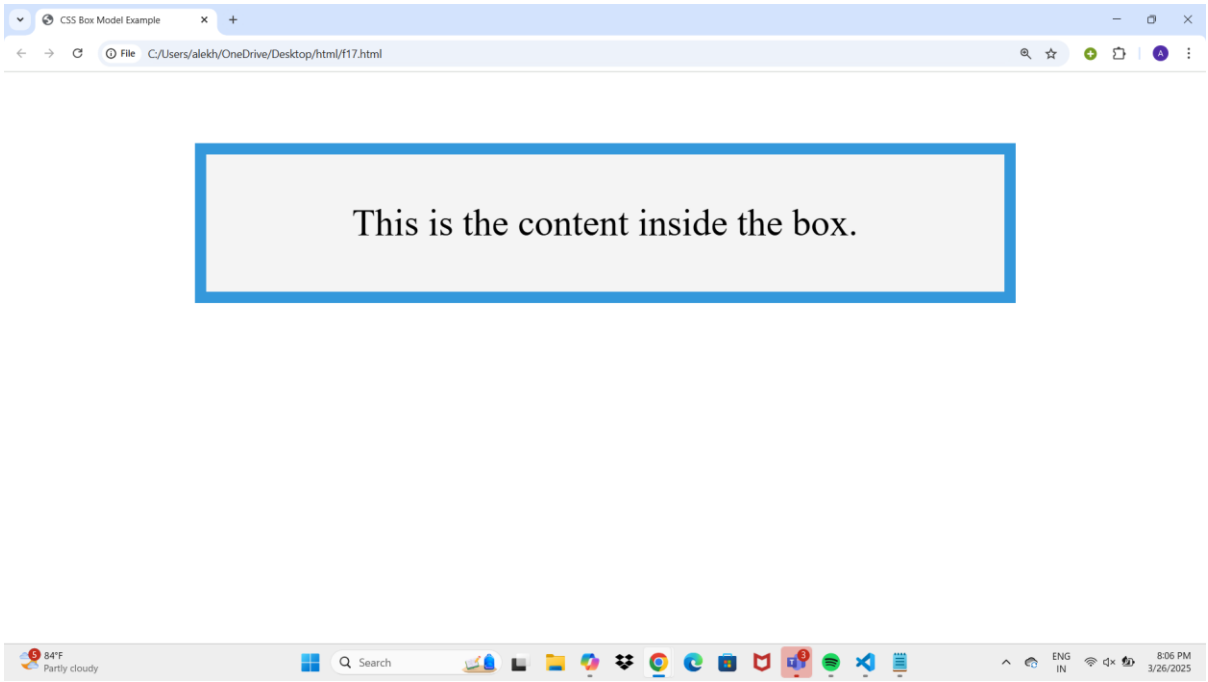
The **Box Model** is critical because it allows developers to control the element's size and spacing precisely.

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>CSS Box Model Example</title>
    <style>
        .box {
            width: 300px;
            background-color: #f4f4f4;
            padding: 20px;
            border: 5px solid #3498db;
            margin: 30px auto;
```

```
            text-align: center;
        }
    </style>
</head>
<body>
    <div class="box">
        This is the content inside the box.
    </div>
</body>
</html>
```

OUTPUT:



6. Applying JavaScript - internal and external, I/O, Type Conversion

a. Write a program to embed internal and external JavaScript in a web page.

A) HTML Code:

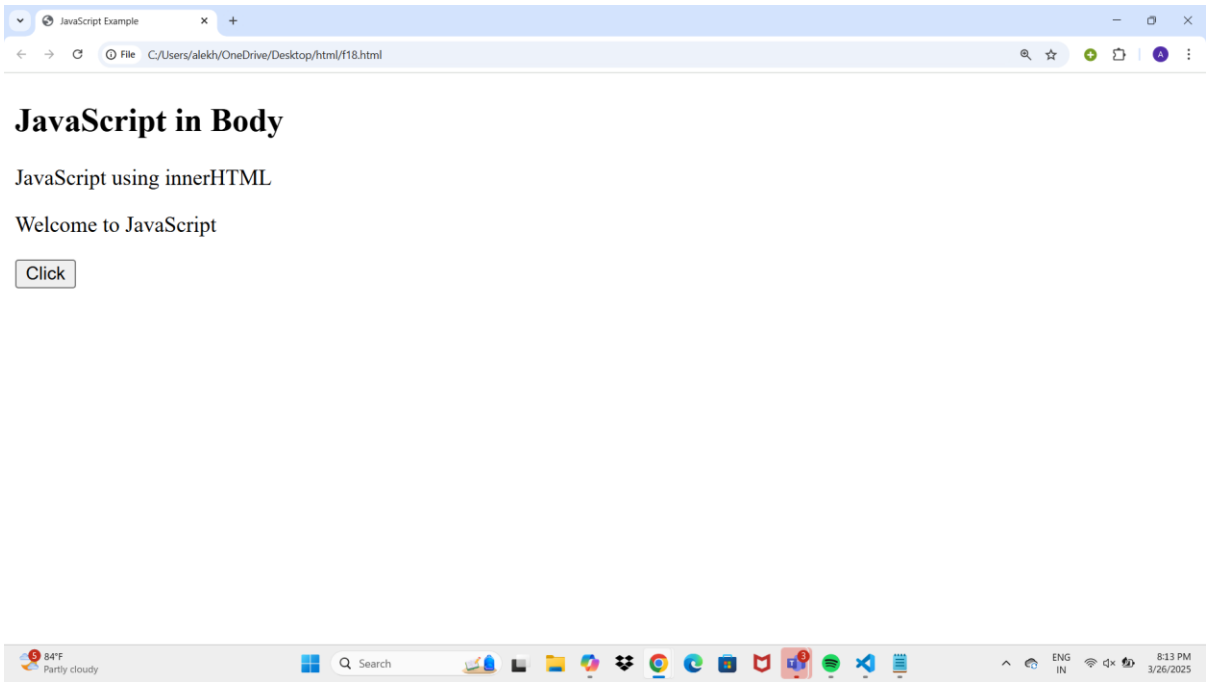
```
<!DOCTYPE html>
```

```
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>JavaScript Example</title>
    <script src="text.js"></script>
</head>
<body>
    <h2>JavaScript in Body</h2>
    <p id="demo"></p>
    <script>
        document.getElementById("demo").innerHTML = "JavaScript using innerHTML";
    </script>
    <p>Welcome to JavaScript</p>
    <form>
        <input type="button" value="Click" onclick="msg()">
    </form>
</body>
</html>
```

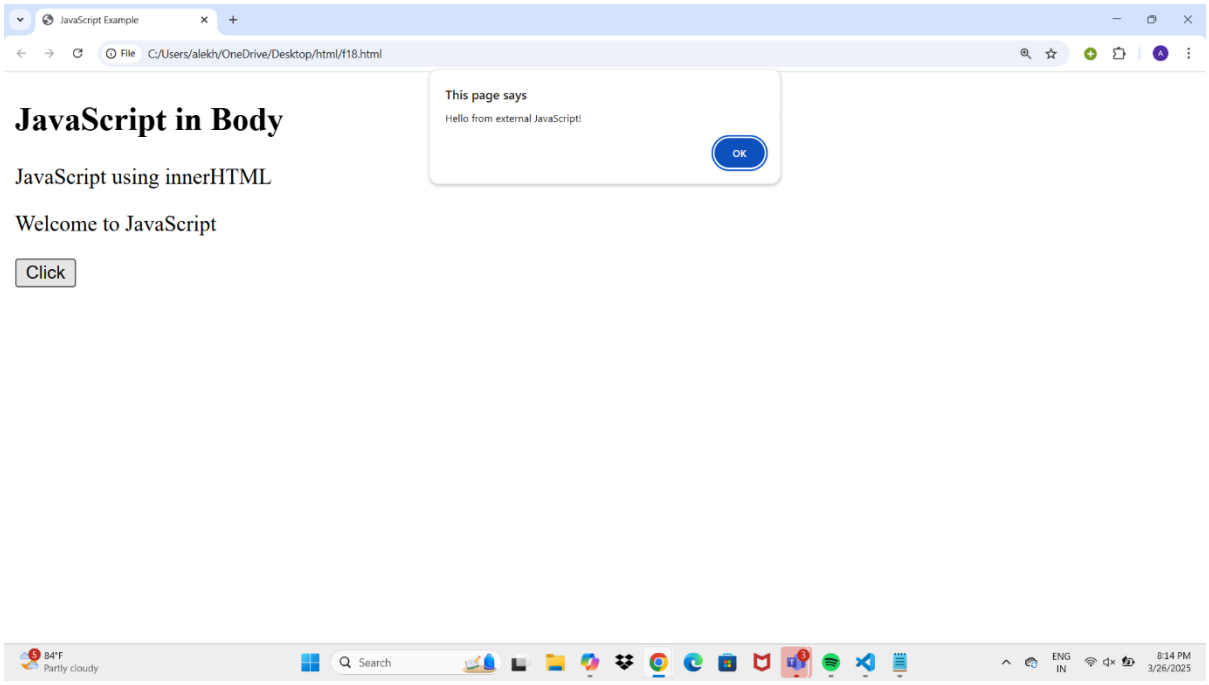
External JavaScript File (text.js):

```
function msg() {
    alert("Hello from external JavaScript!");
}
```

OUTPUT:



After Clicking on button named Click Then output is as follows:



b. Write a program to explain the different ways for displaying output.

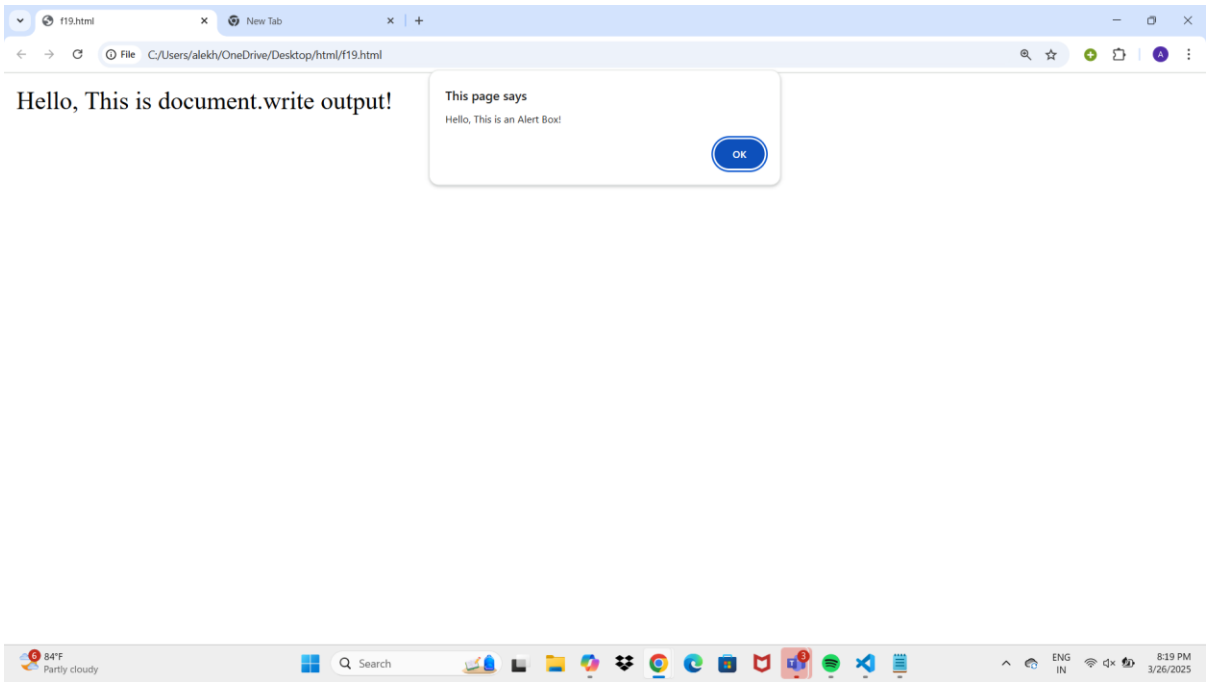
A) HTML Code:

```
<!DOCTYPE html>
<html>
<head>
   <title>JavaScript Output Methods</title>
</head>
<body>
   <h1>JavaScript Output</h1>
   <button onclick="displayAlert()">Click for Alert</button>
   <p id="demo"></p>
   <script>
      console.log("Console Output");
      function displayAlert() {
         alert("Hello, This is an Alert Box!");
         document.write("Hello, This is document.write output!");
         document.getElementById("demo").innerHTML = "Hello, This is innerHTML output!";
      }
      function printPage() {
         window.print();
      }
   </script>
   <button onclick="printPage()">Print Page</button>
</body>
```

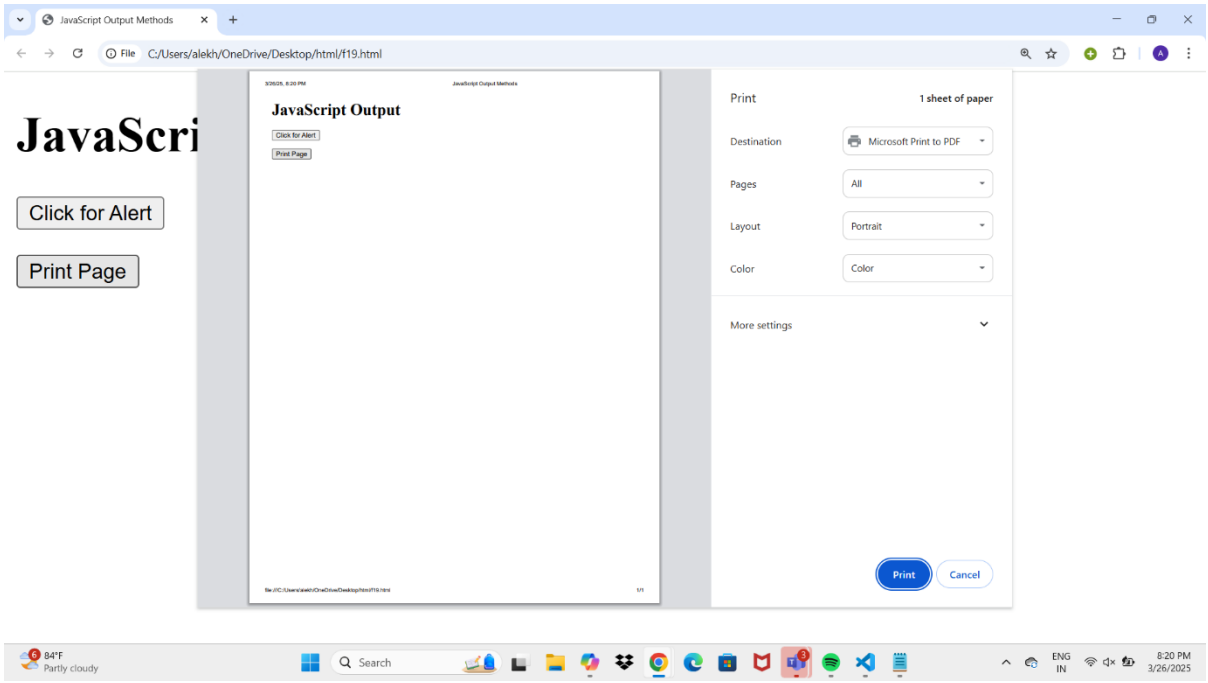
</html>

OUTPUT:

After Clicking on Alert:



After Clicking on Print Page:



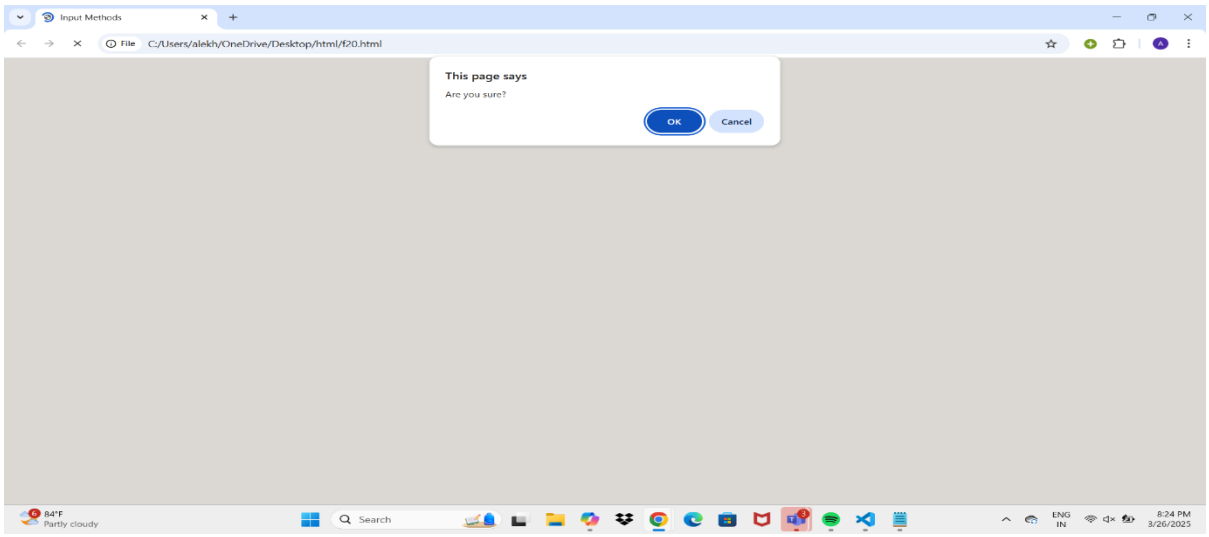
c. Write a program to explain the different ways for taking input.

A) HTML Code:

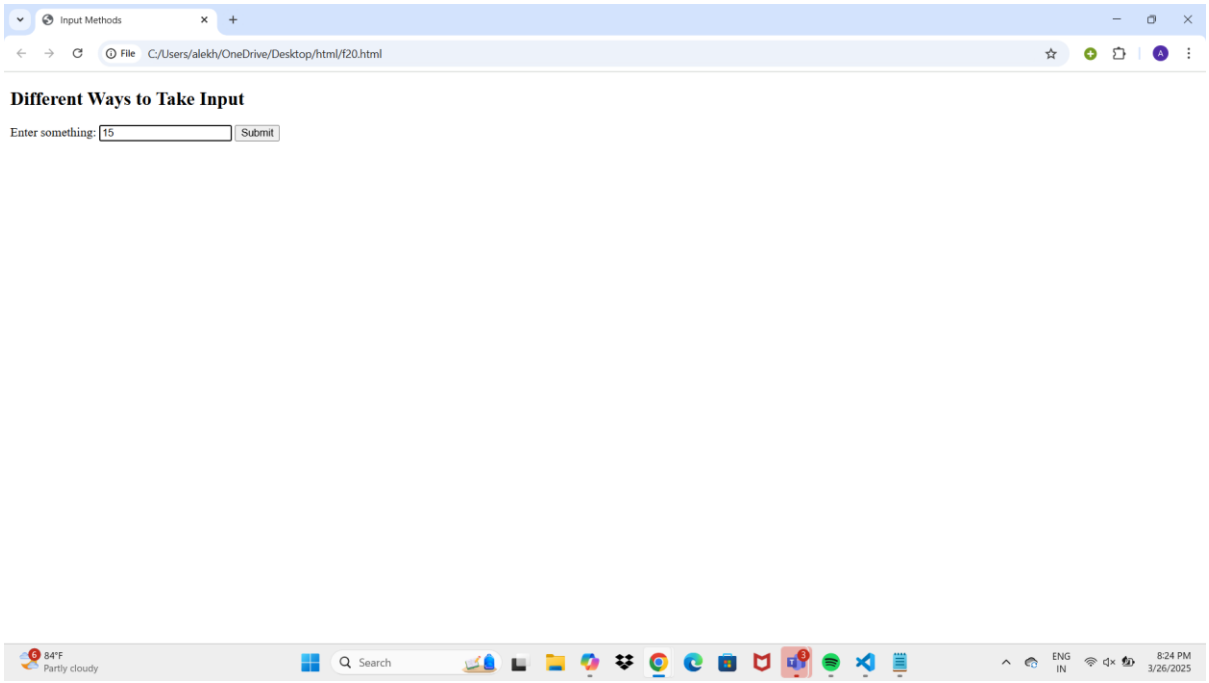
```
<!DOCTYPE html>
<html>
<head>
    <title>Input Methods</title>
</head>
<body>
    <h2>Different Ways to Take Input</h2>
    <script>
        let name = prompt("Enter your name:");
        console.log("Name: " + name);
        let isOkay = confirm("Are you sure?");
        console.log("Confirmation: " + isOkay);
```

```
        function showInput() {
            let userInput = document.getElementById("userInput").value;
            alert("Your Input: " + userInput);
        }
    </script>
    <form>
        <label>Enter something:</label>
        <input type="text" id="userInput">
        <button type="button" onclick="showInput()">Submit</button>
    </form>
</body>
</html>
```

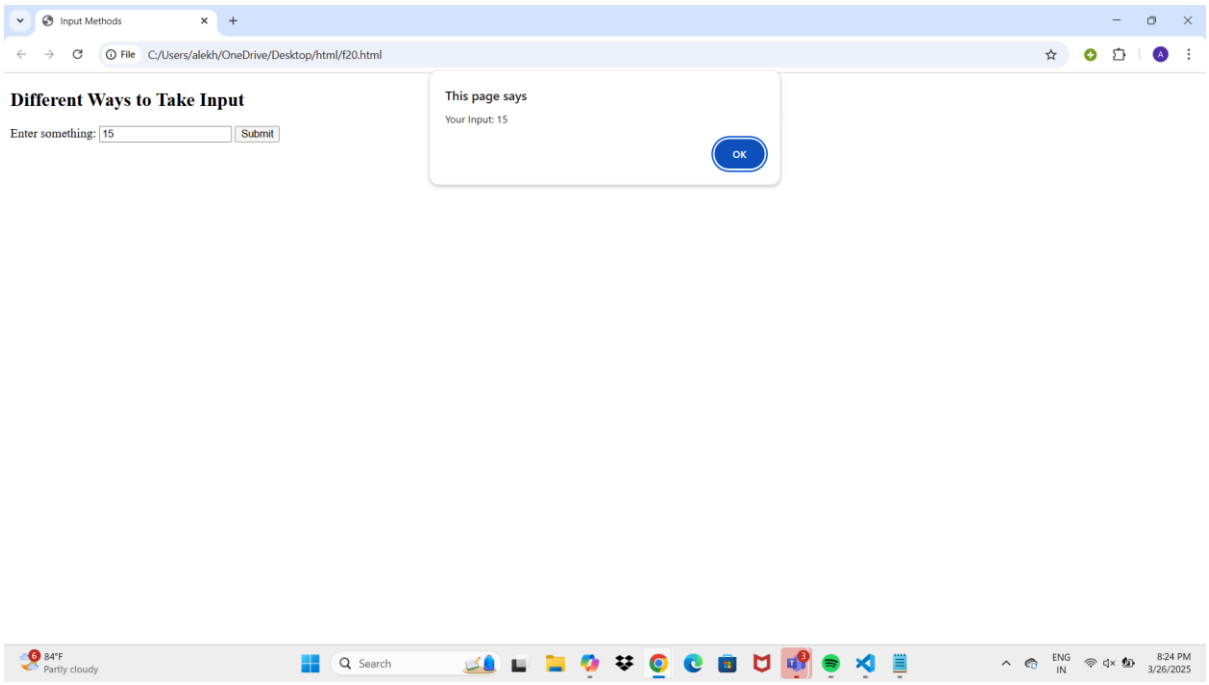
OUTPUT:



After Clicking Ok:



After Clicking on Submit:



d. Create a webpage which uses prompt dialogue box to ask a voter for his name and age. Display the information in table format along with either the voter can vote or not

A) HTML Code:

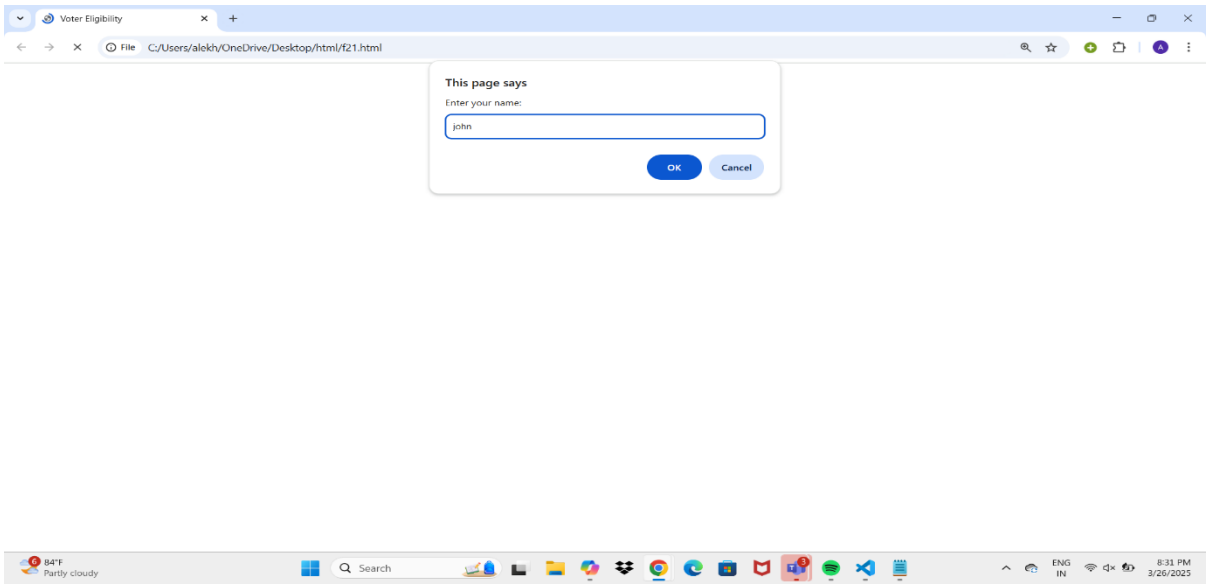
<!DOCTYPE html>

<html>

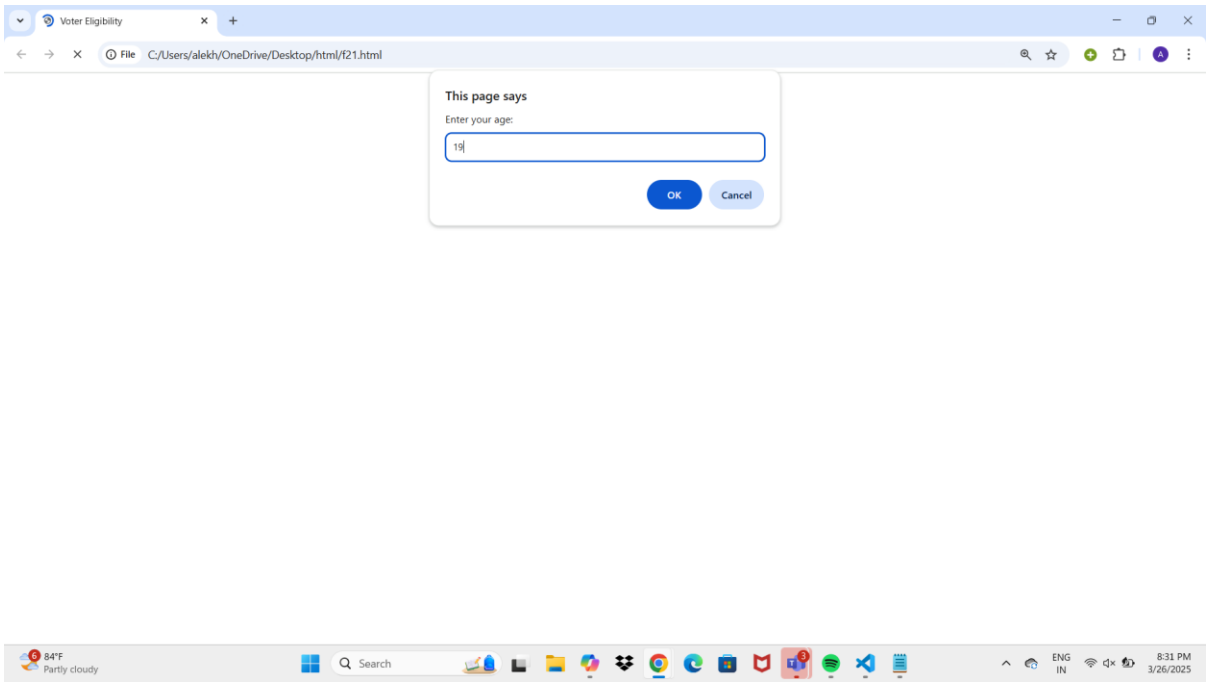
```
<head>
    <title>Voter Eligibility</title>
</head>
<body>
    <h2>Voter Eligibility Checker</h2>
    <script>
        let name = prompt("Enter your name:");
        let age = parseInt(prompt("Enter your age:"));
        let eligibility = (age >= 18) ? "Eligible to Vote" : "Not Eligible to Vote";
        document.write(`
            <table border="1" cellpadding="10">
                <tr>
                    <th>Name</th>
                    <th>Age</th>
                    <th>Eligibility</th>
                </tr>
                <tr>
                    <td>${name}</td>
                    <td>${age}</td>
                    <td>${eligibility}</td>
                </tr>
            </table>
        `);
    </script>
```

```
</body>

</html>
```

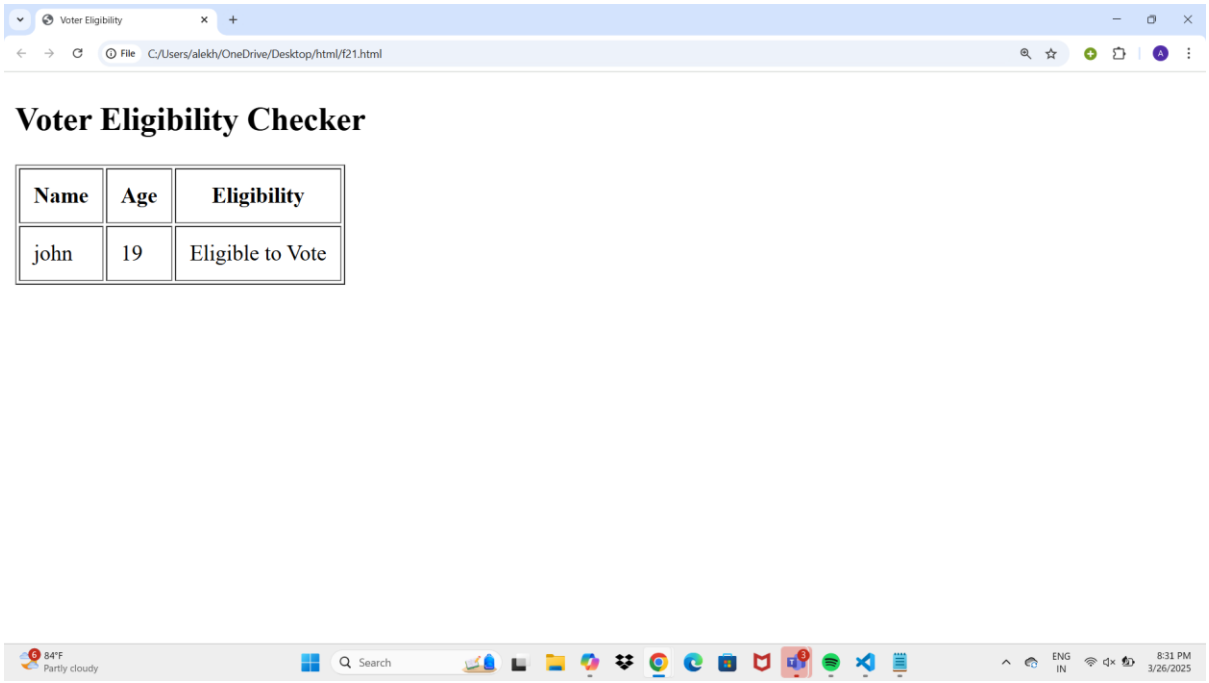
OUTPUT:



After Clicking Ok After entering name in it:



After Clicking Ok After entering age in it Then Output is as follows:



7. Java Script Pre-defined and User-defined Objects

a. Write a program using document object properties and methods

A) Description:

The use of **Document Object Model (DOM)** properties and methods.

1. **Document Properties**:
    a. document.title: Retrieves the title of the webpage, as defined in the <title> tag.

b. document.URL: Provides the current webpage URL.

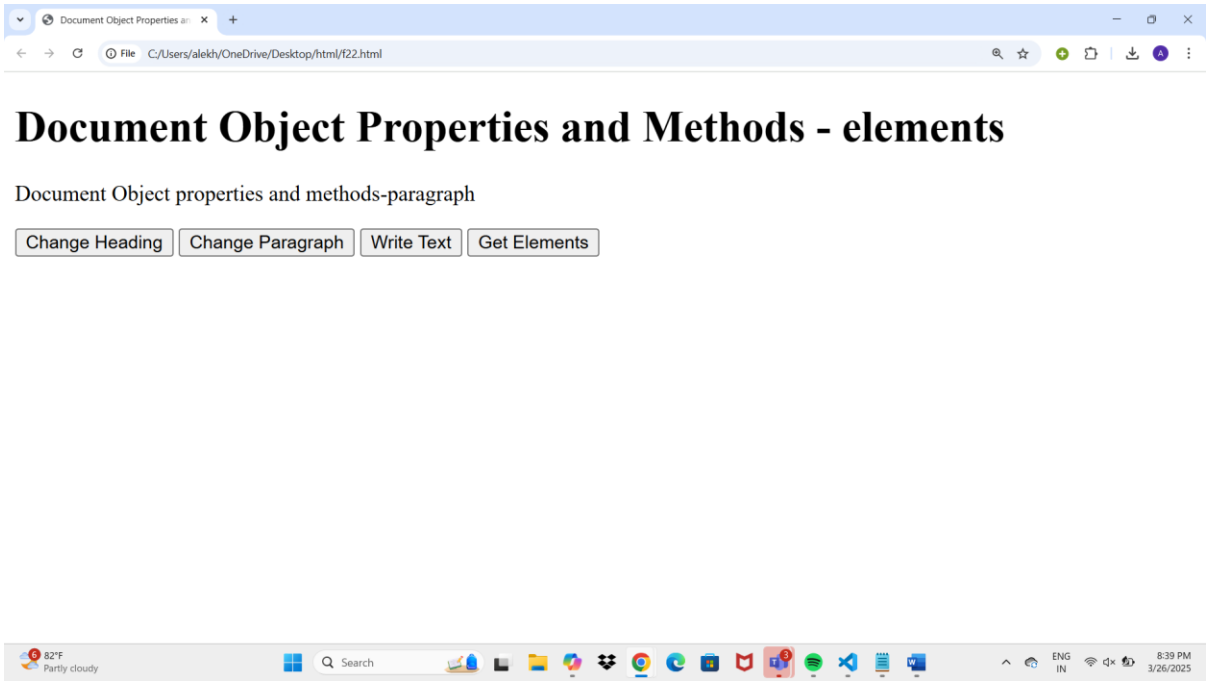
2. **Interactive Methods**:
    a. **Change Heading**: Updates the text inside the <h1> element with id="heading" using innerHTML.
    b. **Change Paragraph**: Alters the text of the <p> element with id="paragraph", also using innerHTML.
    c. **Write Text**: Dynamically writes "Hello, World!" to the webpage using document.write().
    d. **Get Elements**:
        i. Retrieves the <h1> element using document.getElementById().
        ii. Collects all <p> elements using document.getElementsByTagName().
        iii. Finds the <h1> element using a query selector (document.querySelector()).

HTML Code:

```
<!DOCTYPE html>

<html>

<head>

   <title>Document Object Properties and Methods - title</title>

</head>

<body>

   <h1 id="heading">Document Object Properties and Methods - elements</h1>

   <p id="paragraph">Document Object properties and methods-paragraph</p>f

   <button onclick="changeHeading()">Change Heading</button>
```

```
    <button onclick="changeParagraph()">Change Paragraph</button>
    <button onclick="writeText()">Write Text</button>
    <button onclick="getElements()">Get Elements</button>
    <script>
        console.log(document.title);
        console.log(document.URL);
        function changeHeading() {
            document.getElementById("heading").innerHTML = "New Heading";
        }
        function changeParagraph() {
            document.getElementById("paragraph").innerHTML = "New
Paragraph";
        }
        function writeText() {
            document.write("Hello, World!");
        }
        function getElements() {
            console.log(document.getElementById("heading"));
            console.log(document.getElementsByTagName("p"));
            console.log(document.querySelector("#heading"));
        }
    </script> </body>
</html>
```

OUTPUT:



b. Write a program using window object properties and methods.

A) Description:

The usage of **Window Object Properties and Methods** in JavaScript:

1. **Window Object Methods**:
    a. **window.open()**: Opens a new browser window with a specified URL. Here, it opens Google's homepage in a new tab with specified dimensions (width=400, height=300).

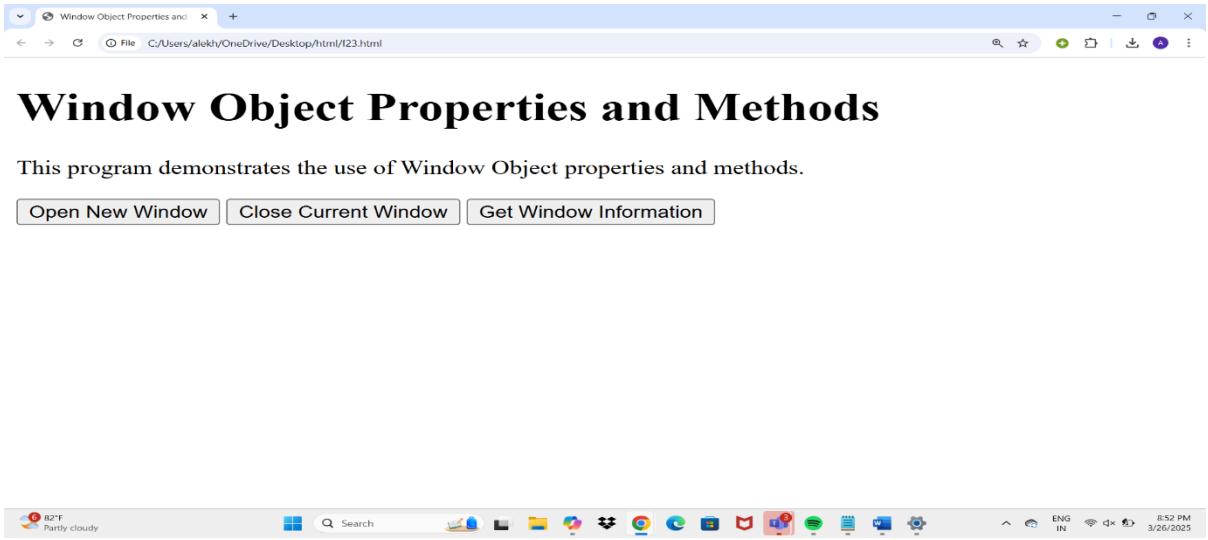
    b. **window.close()**: Closes the current browser window. Note: This works only for windows opened by window.open() due to browser security restrictions.
2. **Window Object Properties**:
    a. **window.innerWidth**: Retrieves the viewport's inner width in pixels.
    b. **window.innerHeight**: Retrieves the viewport's inner height in pixels.
    c. **window.screenX**: Returns the x-coordinate of the window relative to the screen.
    d. **window.screenY**: Returns the y-coordinate of the window relative to the screen.
    e. **window.location**: Provides the URL of the current window.
3. **Interactivity through Buttons**:
    a. **"Open New Window"**: Triggers the openWindow() function to open Google's homepage.
    b. **"Close Current Window"**: Executes the closeWindow() function to attempt closing the window.
    c. **"Get Window Information"**: Calls getInfo() to display window dimensions, coordinates, and URL through alerts.

HTML Code:

```
<!DOCTYPE html>

<html>

<head>

   <title>Window Object Properties and Methods</title>

</head>

<body>

   <h1 id="heading">Window Object Properties and Methods</h1>

   <p id="paragraph">This program demonstrates the use of Window Object properties and methods.</p>
```

```
    <button onclick="openWindow()">Open New Window</button>
    <button onclick="closeWindow()">Close Current Window</button>
    <button onclick="getInfo()">Get Window Information</button>
    <script>
        function openWindow() {
            window.open("https://www.google.com", "_blank", "width=400, height=300");
        }
        function closeWindow() {
            window.close();
        }
        function getInfo() {
            alert("Window Inner Width: " + window.innerWidth);
            alert("Window Inner Height: " + window.innerHeight);
            alert("Window Screen X: " + window.screenX);
            alert("Window Screen Y: " + window.screenY);
            alert("Window Location: " + window.location);
        }
    </script>
</body>
</html>
```

OUTPUT:



c. Write a program using array object properties and methods.

A) Description:

Array Object Properties

1. **Length**:
    a. The length property provides the total number of elements in the array. It is useful for determining the size of the array, tracking data, or performing iterative operations over all elements.

Array Object Methods

1. **Push**:
    a. The push() method is used to add one or more elements to the end of the array. It modifies the array and increases its length. It is often used in dynamic scenarios where new data needs to be appended.
2. **Sort**:
    a. The sort() method arranges the array elements in ascending order by default. It is useful for organizing data and performing operations that require ordered lists. For strings, sorting is straightforward, but for numerical values, adjustments are made using a custom comparison mechanism.

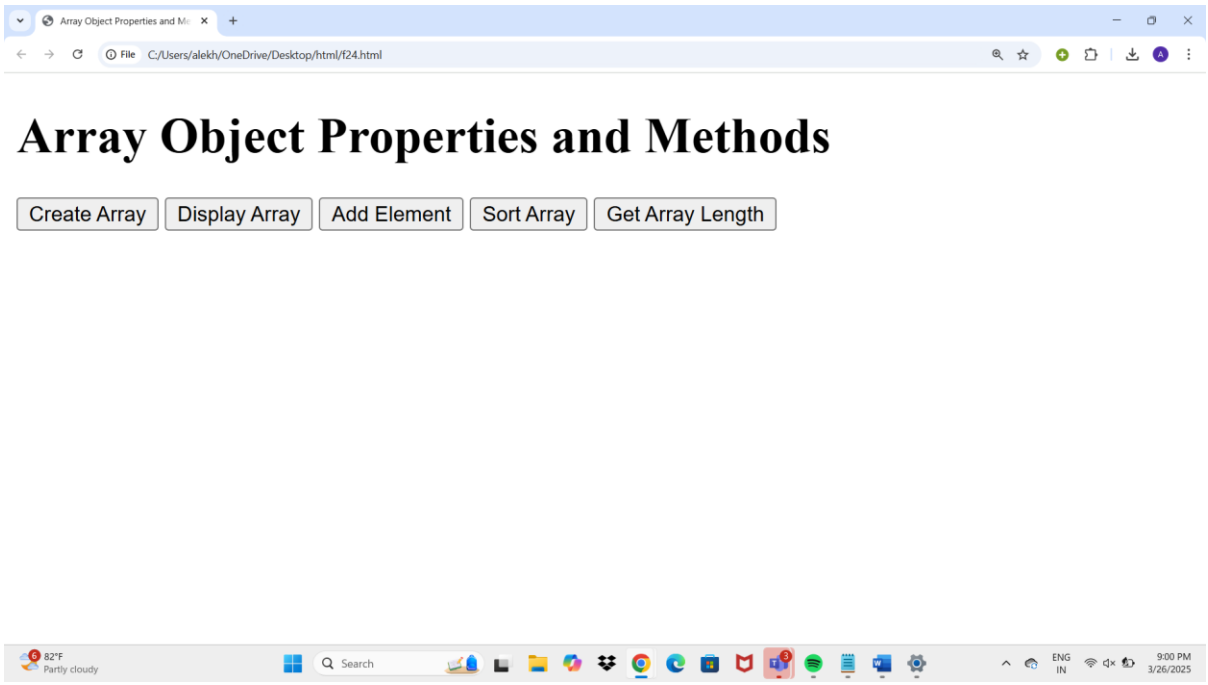
HTML Code:

```
<!DOCTYPE html>
<html>
<head>
    <title>Array Object Properties and Methods</title>
</head>
<body>
    <h1>Array Object Properties and Methods</h1>
    <button onclick="createArray()">Create Array</button>
    <button onclick="displayArray()">Display Array</button>
    <button onclick="addElement()">Add Element</button>
    <button onclick="sortArray()">Sort Array</button>
```

```
    <button onclick="getLength()">Get Array Length</button>
    <div id="result"></div>
    <script>
        let myArray = [];
        function createArray() {
            myArray = [10, 20, 30, 40, 50];
            document.getElementById("result").innerHTML = "Array created: " + myArray;
        }
        function displayArray() {
            document.getElementById("result").innerHTML = "Array: " + myArray;
        }
        function addElement() {
            let element = parseInt(prompt("Enter an element to add:"));
            myArray.push(element);
            document.getElementById("result").innerHTML = "Element added: " + myArray;
        }
        function sortArray() {
            myArray.sort((a, b) => a - b);
            document.getElementById("result").innerHTML = "Array sorted: " + myArray;
        }
        function getLength() {
            document.getElementById("result").innerHTML = "Array length: " + myArray.length;
```

```
    }
  </script>
</body>
</html>
```

OUTPUT:



d. Write a program using math object properties and methods.

A) Description:

Math Properties

These are constants built into JavaScript's Math object that provide essential mathematical values:

- **Math.E**: The base of the natural logarithm, approximately 2.718.
- **Math.PI**: The ratio of a circle's circumference to its diameter, approximately 3.14159.
- **Math.LN2**: The natural logarithm of 2, approximately 0.693.
- **Math.LN10**: The natural logarithm of 10, approximately 2.302.
- **Math.LOG2E**: The base-2 logarithm of E, approximately 1.442.
- **Math.SQRT2**: The square root of 2, approximately 1.414.

Math Methods

These are functions provided by the Math object to perform common mathematical operations:

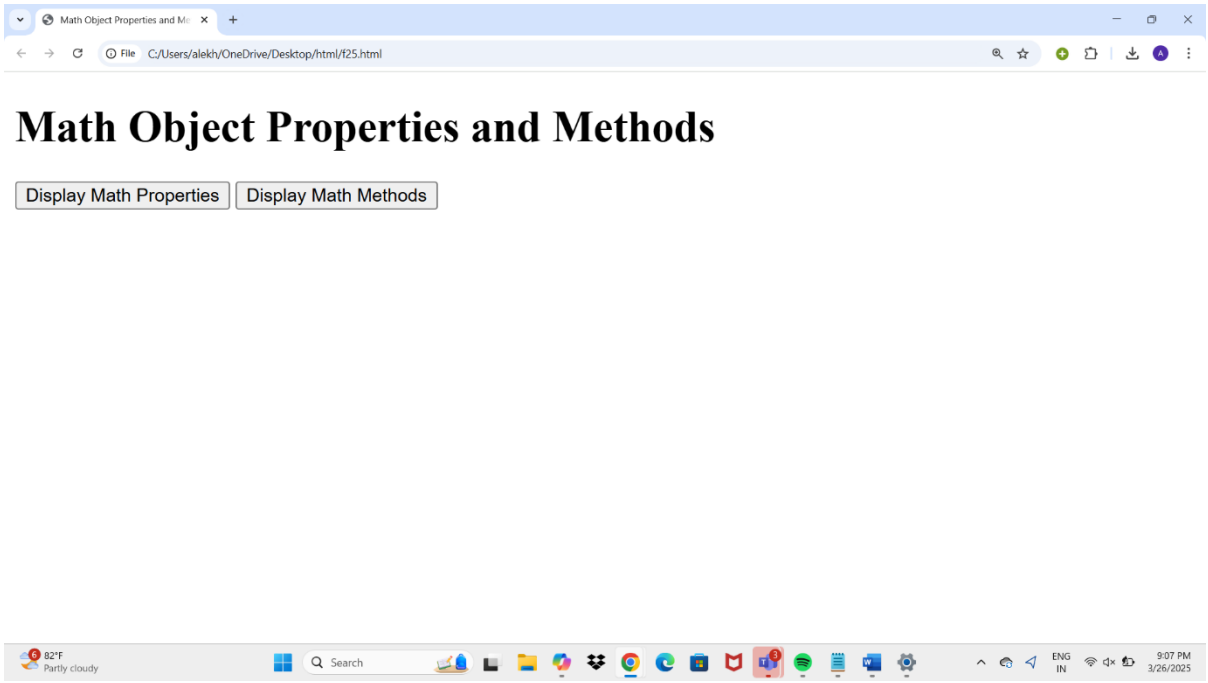
- **Math.log(x)**: Calculates the natural logarithm (base e) of a number x.
- **Math.log2(x)**: Calculates the base-2 logarithm of a number x.
- **Math.sqrt(x)**: Calculates the square root of a number x.
- **Math.max(a, b, ...)**: Finds the largest number among the given arguments.
- **Math.min(a, b, ...)**: Finds the smallest number among the given arguments.

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
    <title>Math Object Properties and Methods</title>
</head> <body>
    <h1>Math Object Properties and Methods</h1>
    <button onclick="displayProperties()">Display Math Properties</button>
    <button onclick="displayMethods()">Display Math Methods</button>
    <div id="result"></div>
    <script>
        function displayProperties() {
```

```
        let result = "";
        result += "Value of E: " + Math.E + "<br>";
        result += "Value of PI: " + Math.PI + "<br>";
        result += "Value of LN2: " + Math.LN2 + "<br>";
        result += "Value of LN10: " + Math.LN10 + "<br>";
        result += "Value of LOG2E: " + Math.LOG2E + "<br>";
        result += "Value of SQRT2: " + Math.SQRT2 + "<br>";
        document.getElementById("result").innerHTML = result;
    }
    function displayMethods() {
        let result = "";
        result += "Natural Log of 10: " + Math.log(10) + "<br>";
        result += "Base-2 Logarithm of 10: " + Math.log2(10) + "<br>";
        result += "Square Root of 16: " + Math.sqrt(16) + "<br>";
        result += "Maximum of 10 and 20: " + Math.max(10, 20) + "<br>";
        result += "Minimum of 10 and 20: " + Math.min(10, 20) + "<br>";
        document.getElementById("result").innerHTML = result;
    } </script>
</body>
</html>
```

OUTPUT:



e. Write a program using string object properties and methods.

A) Description:

String properties like are fundamental for tasks involving validation, while string methods empower developers to manipulate, search, and format strings efficiently. They're essential for creating dynamic applications that handle text data, such as messaging apps, word processors, and even websites.

String Properties

- **`length`**: Returns the length of the string, which includes the number of characters in it (including spaces, punctuation, and special characters).

String Methods

- **`toUpperCase()`**: Converts all characters in the string to uppercase (e.g., "hello" becomes "HELLO").

- **`toLowerCase()`**: Converts all characters in the string to lowercase (e.g., "HELLO" becomes "hello").

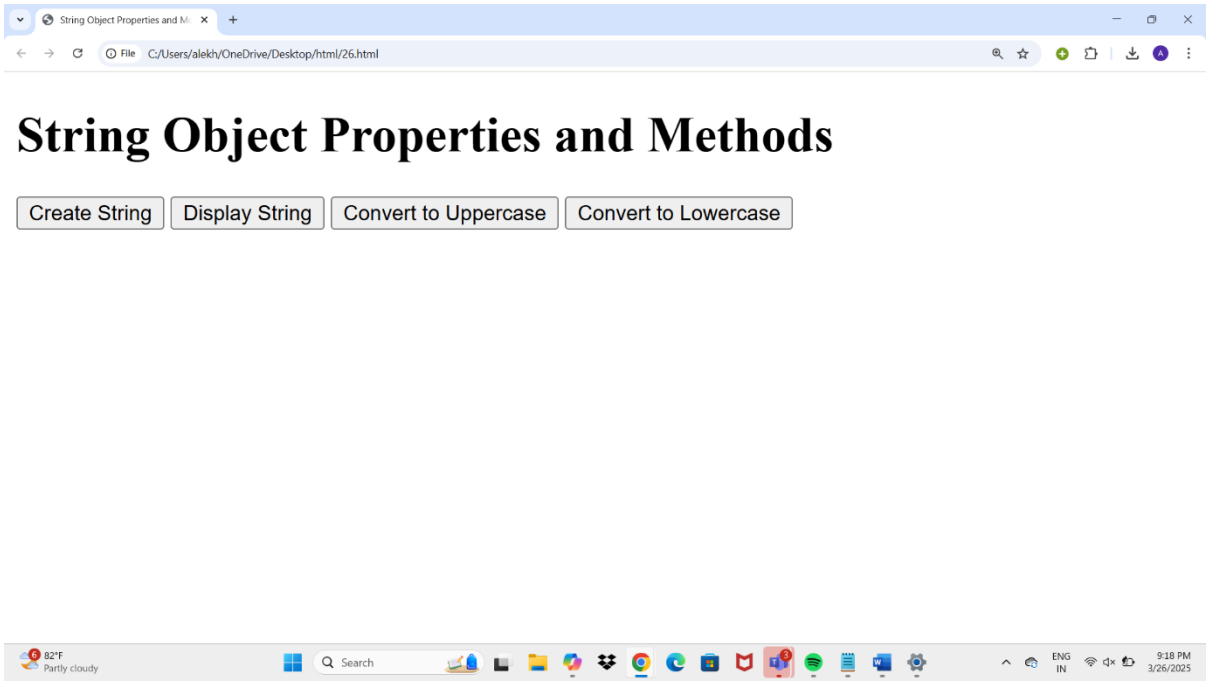
- **CharAt(index):** Retrieves the character at the given index.

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
  <title>String Object Properties and Methods</title>
</head>
<body>
   <h1>String Object Properties and Methods</h1>
   <button onclick="createString()">Create String</button>
   <button onclick="displayString()">Display String</button>
```

```
    <button onclick="convertToUpper()">Convert to Uppercase</button>
    <button onclick="convertToLower()">Convert to Lowercase</button>
    <div id="result"></div>
    <script>
        let myString = "";
        function createString() {
            myString = "Hello, World!";
            document.getElementById("result").innerHTML = "String created: " + myString;
        }
        function displayString() {
            document.getElementById("result").innerHTML = "String: " + myString + "<br>Length: " + myString.length;
        }
        function convertToUpper() {
            document.getElementById("result").innerHTML = "Uppercase: " + myString.toUpperCase();
        }
        function convertToLower() {
            document.getElementById("result").innerHTML = "Lowercase: " + myString.toLowerCase();
        }
    </script>
</body>
</html>
```

OUTPUT:



f. Write a program using regex object properties and methods.

A) Description:

Regex Properties

1. **pattern**:
    a. Refers to the regex expression itself, which defines the sequence of characters to match.
    b. Typically written between forward slashes (/pattern/) or as a constructor (new RegExp(pattern)).

Regex Methods

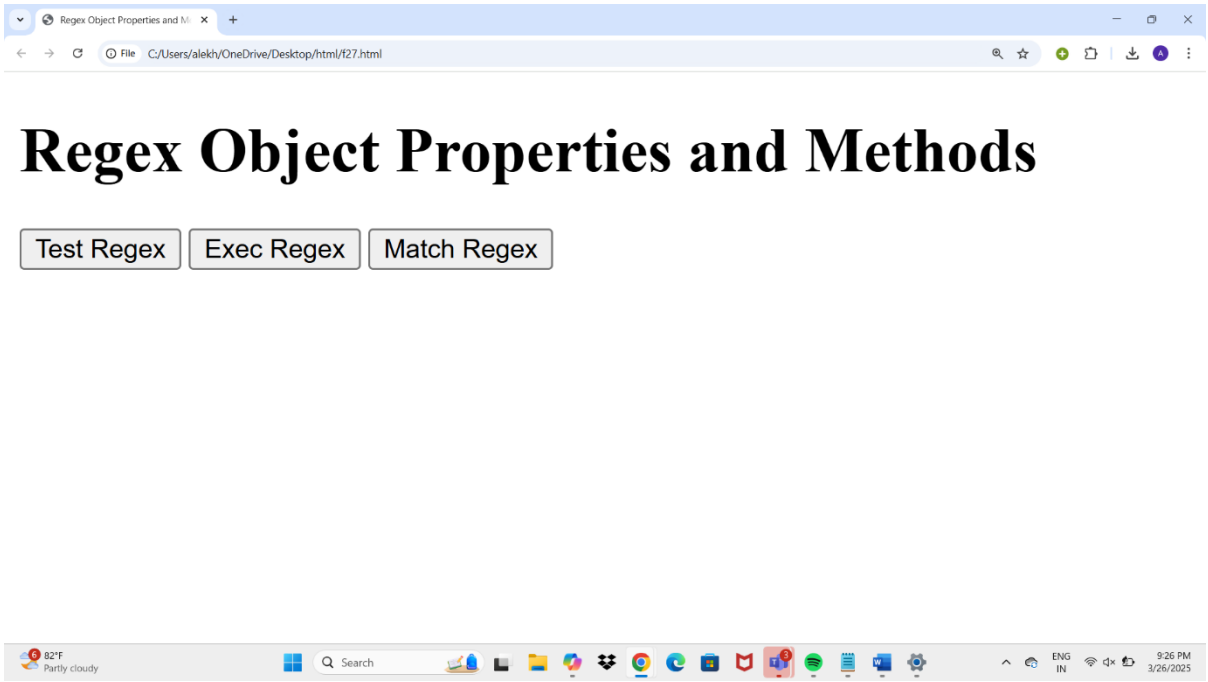
1. **test(string)**:
    a. Tests whether the regex matches any part of the string.
    b. Returns a boolean value: true if a match is found, otherwise false.
2. **exec(string)**:
    a. Executes the regex on the given string.
    b. Returns an array containing the matched text and capturing groups if found; otherwise, returns null.
3. **match()**:
    a. Used on a string to apply the regex and return all matches as an array.
    b. If no match is found, it returns null.

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
  <title>Regex Object Properties and Methods</title>
</head>
<body>
  <h1>Regex Object Properties and Methods</h1>
  <button onclick="testRegex()">Test Regex</button>
  <button onclick="execRegex()">Exec Regex</button>
  <button onclick="matchRegex()">Match Regex</button>
  <div id="result"></div>
  <script>
    let pattern = /hello/gi;
    let str = "Hello, hello, HELLO!";
    function testRegex() {
      let result = pattern.test(str);
```

```
        document.getElementById("result").innerHTML = "Test result: " +
result;
    }
    function execRegex() {
        let result = pattern.exec(str);
        document.getElementById("result").innerHTML = "Exec result: " +
result;
    }
    function matchRegex() {
        let result = str.match(pattern);
        document.getElementById("result").innerHTML = "Match result: " +
result;
    }
  </script>
</body>
</html>
```

OUTPUT:



g. Write a program using date object properties and methods.

A) Description:

Date Object Properties

The Date object does not have many intrinsic properties directly tied to it, but a few key internal components are used in its methods. It works with specific elements of a date, such as the year, month, day, hour, etc.

Date Object Methods

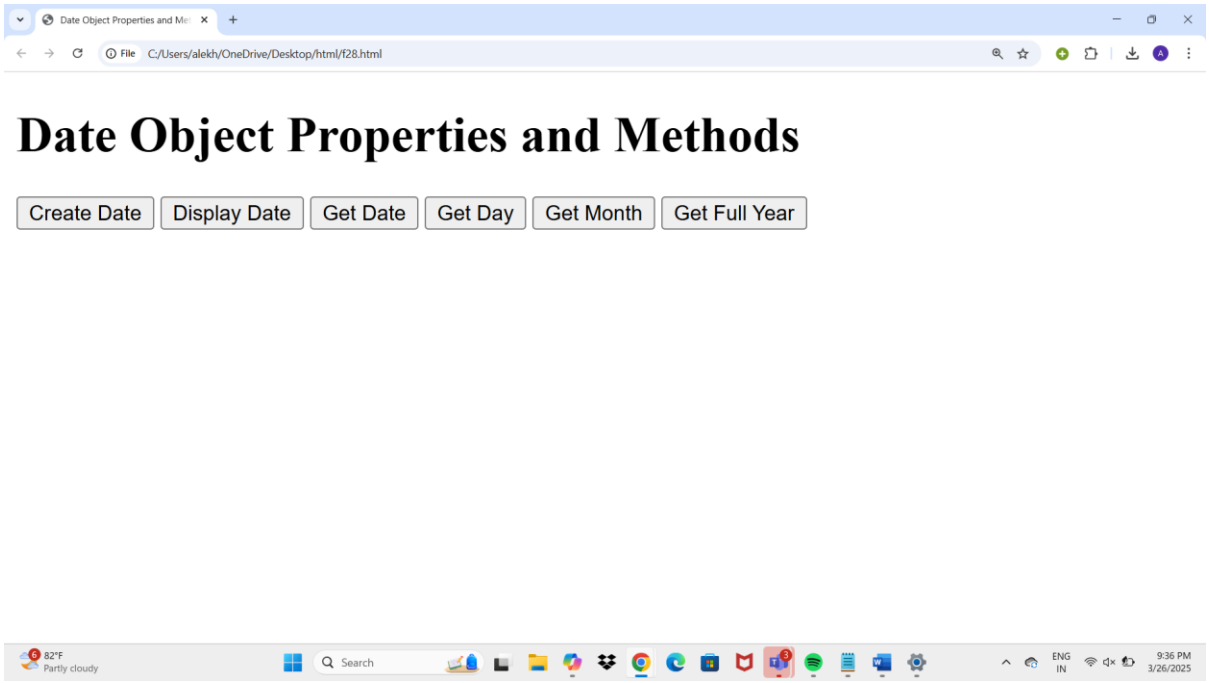
1. **getDate()**:
    a. Retrieves the day of the month (1–31) from the Date object.
2. **getDay()**:
    a. Returns the day of the week as a number (0–6), where 0 represents Sunday and 6 represents Saturday.
3. **getMonth()**:
    a. Retrieves the month of the year as a zero-based value (0–11), where 0 corresponds to January and 11 to December.
4. **getFullYear()**:
    a. Returns the four-digit year of the specified date.

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
    <title>Date Object Properties and Methods</title>
</head>
<body>
    <h1>Date Object Properties and Methods</h1>
    <button onclick="createDate()">Create Date</button>
    <button onclick="displayDate()">Display Date</button>
    <button onclick="getDate()">Get Date</button>
    <button onclick="getDay()">Get Day</button>
    <button onclick="getMonth()">Get Month</button>
    <button onclick="getFullYear()">Get Full Year</button>
    <div id="result"></div>
    <script>
        let date = new Date();
        function createDate() {
            date = new Date();
            document.getElementById("result").innerHTML = "Date created: " +
date;
        }
        function displayDate() {
            document.getElementById("result").innerHTML = "Date: " + date;
        }
```

```
        function getDate() {
            document.getElementById("result").innerHTML = "Date: " + date.getDate();
        }
        function getDay() {
            document.getElementById("result").innerHTML = "Day: " + date.getDay();
        }
        function getMonth() {
            document.getElementById("result").innerHTML = "Month: " + date.getMonth();
        }
        function getFullYear() {
            document.getElementById("result").innerHTML = "Full Year: " + date.getFullYear();
        }
    </script>
</body>
</html>
```

OUTPUT:



h. Write a program to explain user-defined object by using properties, methods, accessors, constructors and display.

A) Description:

1. **Properties**:
    a. These are variables associated with an object that store information relevant to it.
    b. In a user-defined object, properties like `name` and `age` are specific to each instance of the object.
2. **Accessors (Getters and Setters)**:
    a. **Getter**: Retrieves the value of a property in a controlled way.
    b. **Setter**: Updates the value of a property while allowing validation or specific logic.
3. **Constructor**:
    a. A constructor is a special function that initializes the properties of an object when it is created.
    b. It ensures that an object starts with predefined values or structure.
4. **Methods**:
    a. Methods are functions associated with an object, defining its behaviors.
    b. Examples include:
        i. **`display()`**: Presents an object's data in a readable format.
        ii. **`greet()`**: Provides custom interactions or outputs based on the object's properties.

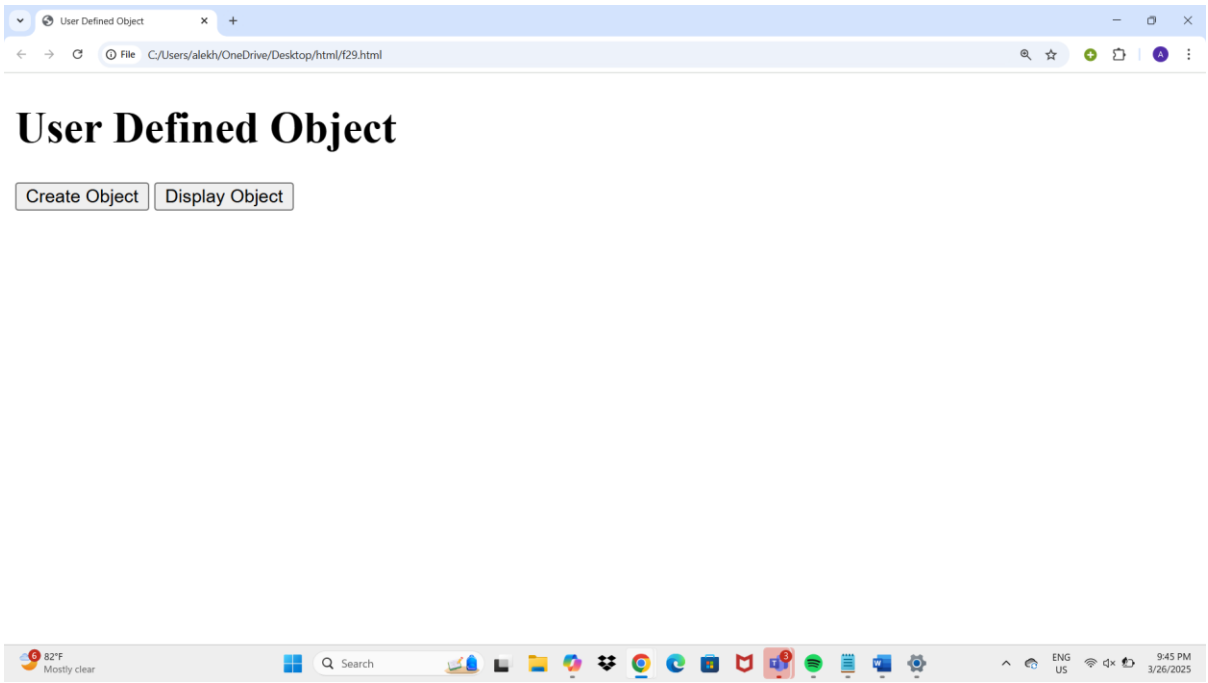
HTML Code:

```
<!DOCTYPE html>
<html>
<head>
<title>User Defined Object</title>
</head>
<body>
<h1>User Defined Object</h1>
<button onclick="createObject()">Create Object</button>
<button onclick="displayObject()">Display Object</button>
<div id="result"></div>
<script>
class Person {
constructor(name, age) {
this._name = name;
this._age = age;
}
```

```
get name() {
return this._name;
}
set name(value) {
this._name = value;
}
get age() {
return this._age;
}
set age(value) {
this._age = value;
}
display() {
return `Name: ${this._name}, Age: ${this._age}`;
}
greet() {
return `Hello, my name is ${this._name} and I am ${this._age} years old.`;
}
}
let person;
function createObject() {
person = new Person("John Doe", 30);
document.getElementById("result").innerHTML = "Object created successfully!";
```

```
}

function displayObject() {

if (person) {

document.getElementById("result").innerHTML = person.display();

} else {

document.getElementById("result").innerHTML = "Please create an object first!";

}

}

</script>

</body>

</html>
```

OUTPUT:



8. Java Script Conditional Statements and Loops

a. Write a program which asks the user to enter three integers, obtains the numbers from the user and outputs HTML text that displays the larger number followed by the words "LARGER NUMBER" in an information message dialog. If the numbers are equal, output HTML text as "EQUAL NUMBERS"

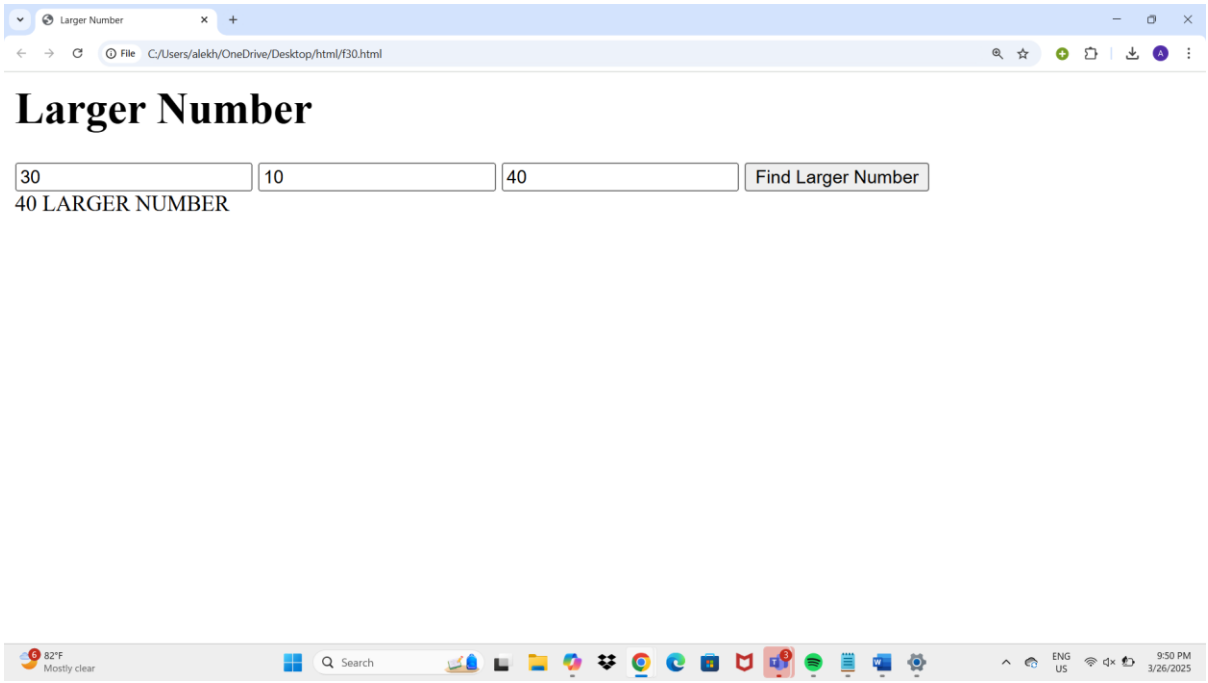
A)

HTML Code:

```
html>
<head>
      <title>Larger Number</title>

</head>
<body>
   <h1>Larger Number</h1>
   <input id="num1" type="number" placeholder="Enter first number">
   <input id="num2" type="number" placeholder="Enter second number">
   <input id="num3" type="number" placeholder="Enter third number">
   <button onclick="findLargerNumber()">Find Larger Number</button>
   <div id="result" ></div>
   <script>
      function findLargerNumber() {
         let num1 = parseInt(document.getElementById("num1").value);
         let num2 = parseInt(document.getElementById("num2").value);
         let num3 = parseInt(document.getElementById("num3").value);
         let largerNum = Math.max(num1, num2, num3);
         if (num1 === num2 && num2 === num3) {
            document.getElementById("result").innerHTML = "EQUAL
NUMBERS";
         } else {
            document.getElementById("result").innerHTML = `${largerNum}
LARGER NUMBER`;
```

```
        }
    }
  </script>
</body>
</html>
```

OUTPUT:



b. Write a program to display week days using switch case.

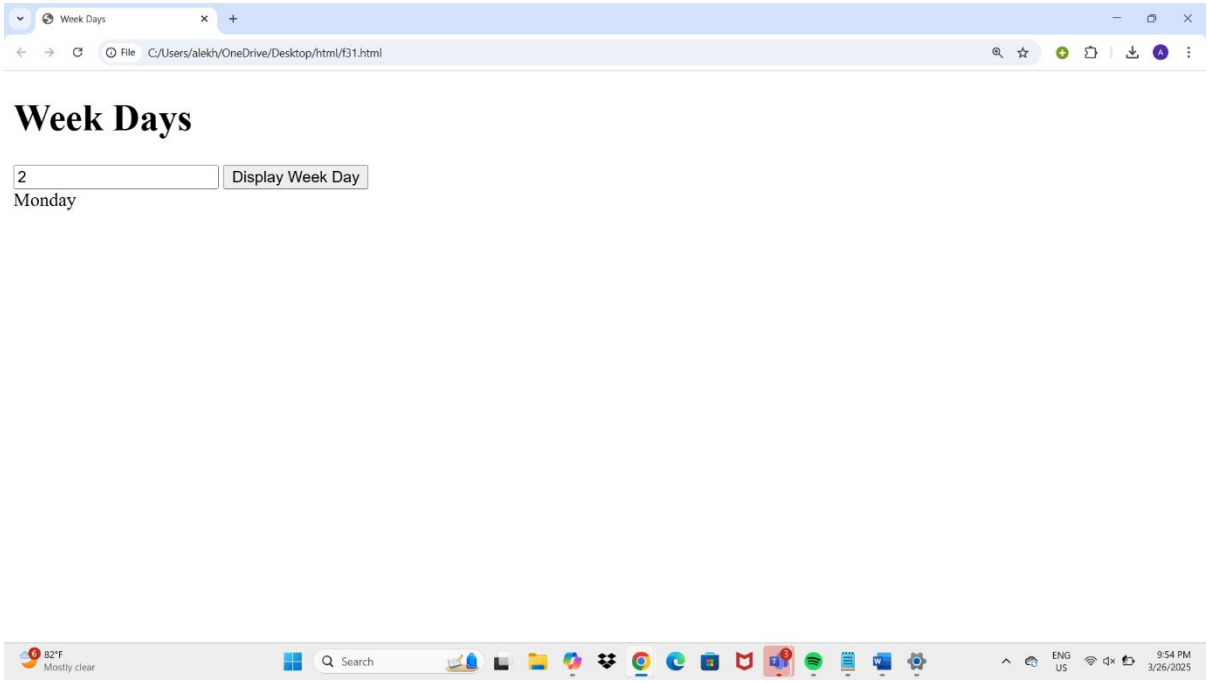
A)

HTML Code:

```
<!DOCTYPE html>
<html>
<head>
   <title>Week Days</title>
</head>
<body>
   <h1>Week Days</h1>
   <input id="day" type="number" placeholder="Enter day number (1-7)">
   <button onclick="displayWeekDay()">Display Week Day</button>
   <div id="result"></div>
   <script>
      function displayWeekDay() {
         let day = parseInt(document.getElementById("day").value);
         let result = document.getElementById("result");
         switch (day) {
            case 1:
               result.innerHTML = "Sunday";
               break;
            case 2:
               result.innerHTML = "Monday";
               break;
            case 3:
               result.innerHTML = "Tuesday";
               break;
            case 4:
               result.innerHTML = "Wednesday";
               break;
            case 5:
               result.innerHTML = "Thursday";
               break;
            case 6:
               result.innerHTML = "Friday";
               break;
            case 7:
```

```
                result.innerHTML = "Saturday";
                break;
            default:
                result.innerHTML = "Invalid day";
        }
    }
  </script>
</body>
</html>
```

OUTPUT:



c. Write a program to print 1 to 10 numbers using for, while and do-while loops.

A)

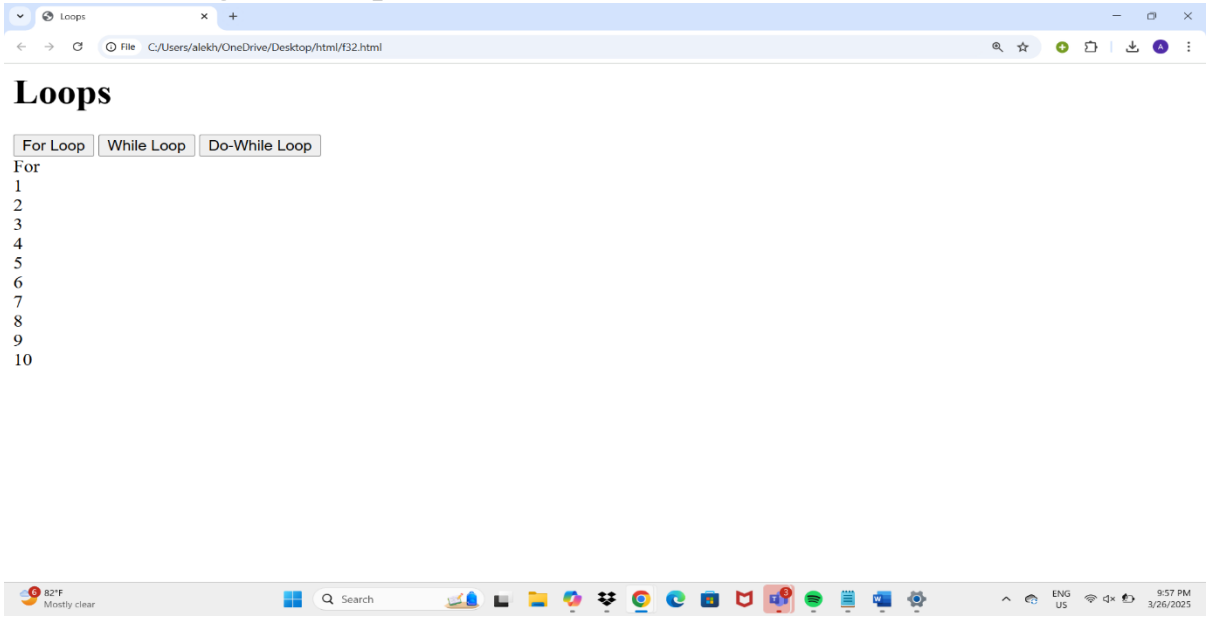
HTML Code:

```
<html>
<head>
	<title>Loops</title>
</head>
<body>
   <h1>Loops</h1>
   <button onclick="printNumbersFor()">For Loop</button>
   <button onclick="printNumbersWhile()">While Loop</button>
   <button onclick="printNumbersDoWhile()">Do-While Loop</button>
   <div id="result"></div>
   <script>
      function printNumbersFor() {
         let result = "For"+`<br>`;

         for (let i = 1; i <= 10; i++) {
            result += `${i}<br>`;
         }
         document.getElementById("result").innerHTML = result;
      }
      function printNumbersWhile() {
         let result = "While"+`<br>`;
         let i = 1;
         while (i <= 10) {
            result += `${i}<br>`;
            i++;
         }
         document.getElementById("result").innerHTML = result;
      }
      function printNumbersDoWhile() {
         let result = "DoWhile"+`<br>`;
         let i = 1;
         do {
            result += `${i}<br>`;
            i++;
         } while (i <= 10);
         document.getElementById("result").innerHTML = result;
      }
   </script>
</body>
```

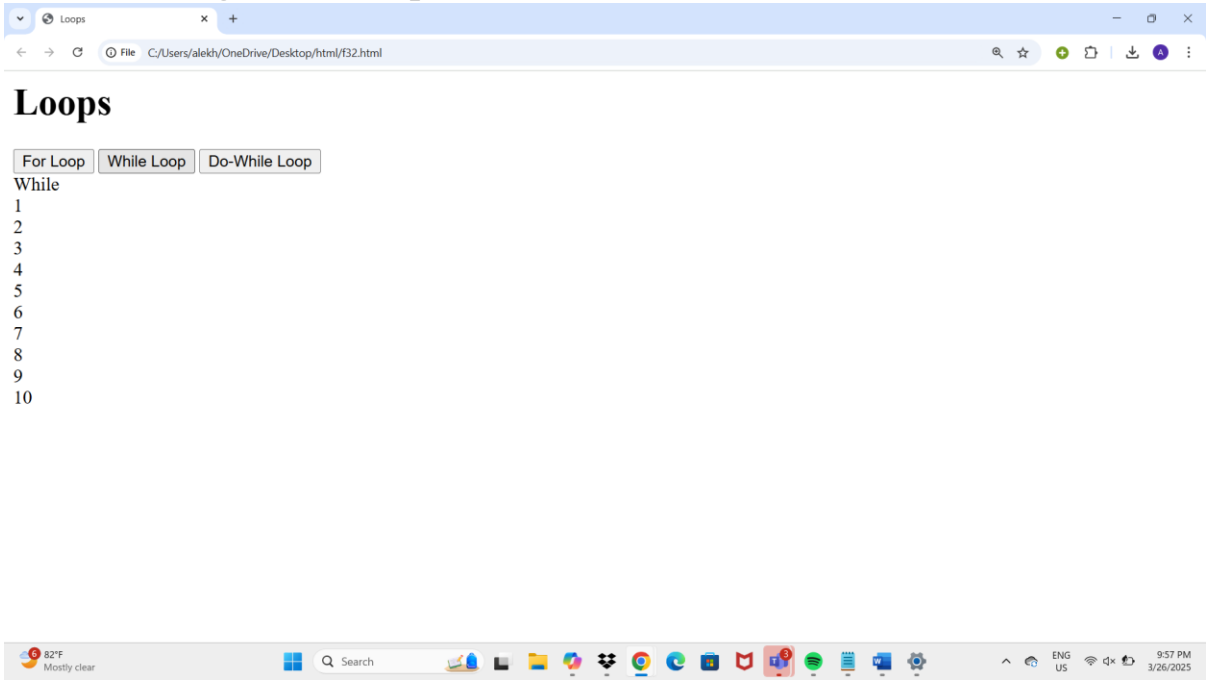
</html>

OUTPUT:

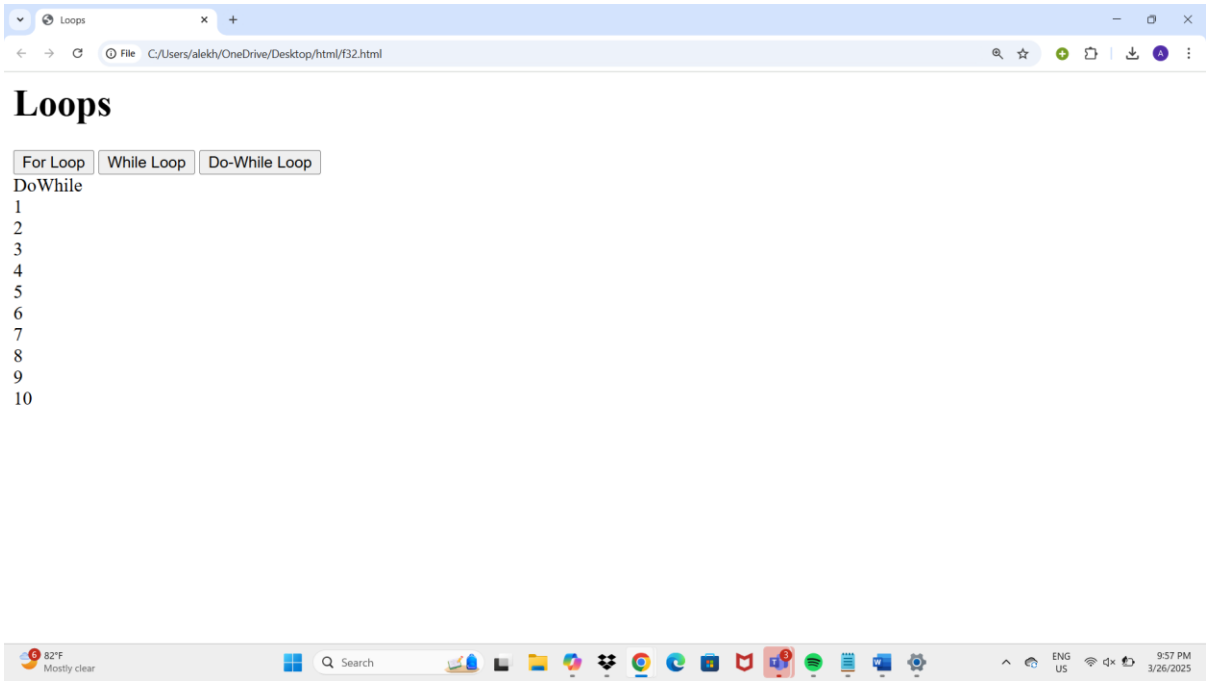
After Clicking For Loop:



After Clicking While Loop:



After Clicking Do While Loop:



d. Write a program to print data in object using for-in, for-each and for-of loops.

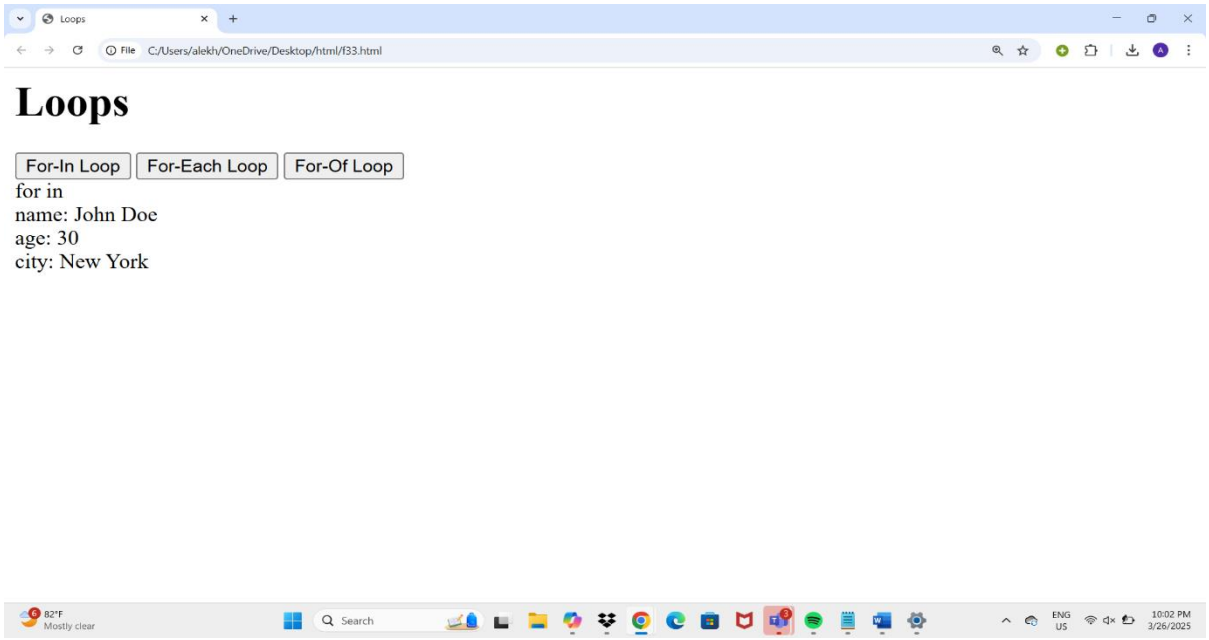
A)

HTML Code:

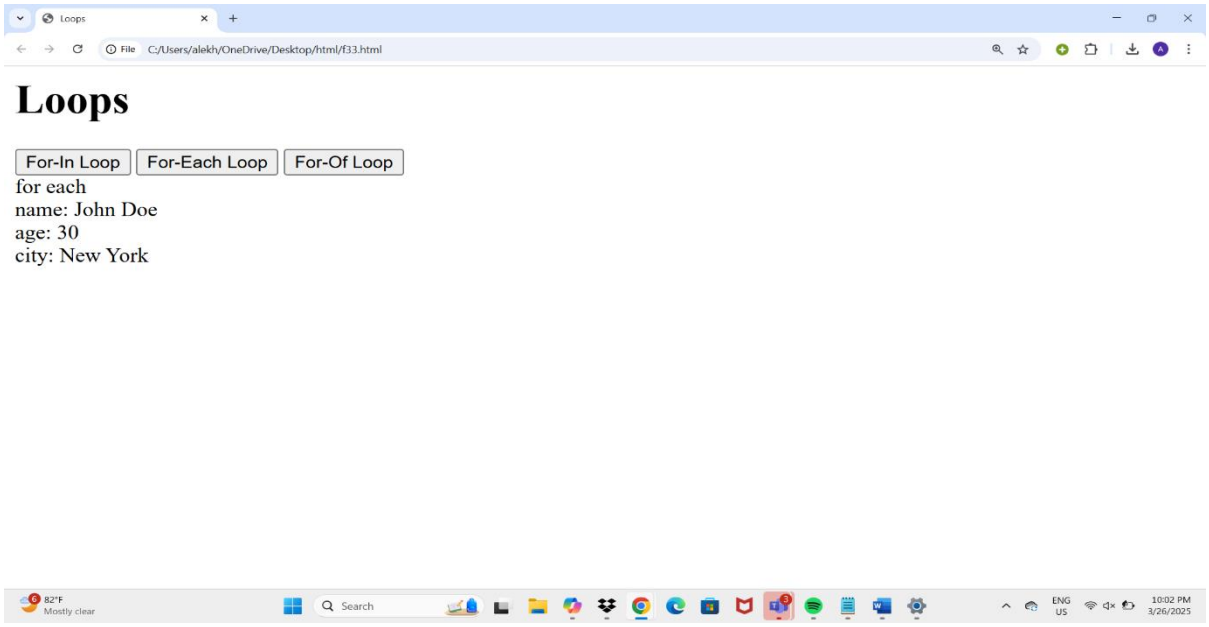
```
<html>
<head>
      <title>Loops</title>
</head>
<body>
   <h1>Loops</h1>
   <button onclick="ForIn()">For-In Loop</button>
   <button onclick="ForEach()">For-Each Loop</button>
   <button onclick="ForOf()">For-Of Loop</button>

   <div id="result"></div>

   <script>
      let person = {
         name: "John Doe",
         age: 30,
         city: "New York"
      };
      function ForIn() {
         let result = "for in"+`<br>`;
         for (let key in person) {
            result += `${key}: ${person[key]}<br>`;
         }
         document.getElementById("result").innerHTML = result;
      }
      function ForEach() {
         let result = "for each"+`<br>`
         Object.keys(person).forEach(key => {
            result += `${key}: ${person[key]}<br>`;
         });
         document.getElementById("result").innerHTML = result;
      }
      function ForOf() {
         let result = "for of"+`<br>`
         for (let [key, value] of Object.entries(person)) {
            result += `${key}: ${value}<br>`;
         }
         document.getElementById("result").innerHTML = result;
      }
   </script>
</body>
</html>
```

OUTPUT:

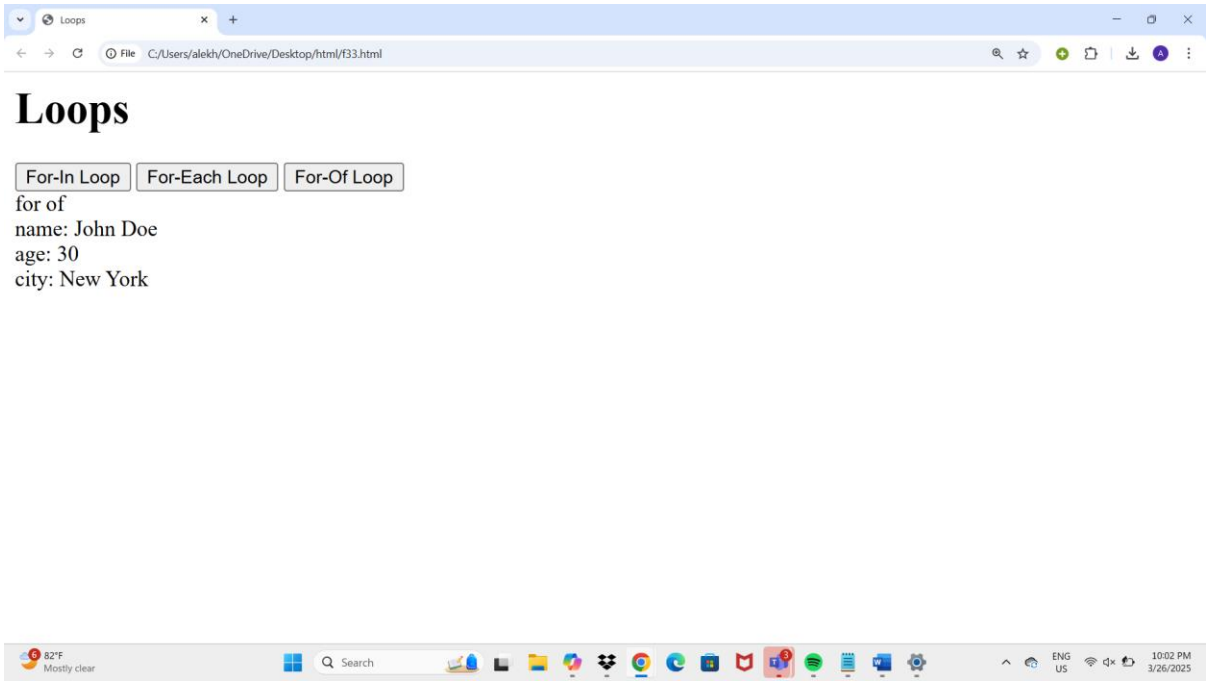
After Clicking For-In Loop:



After Clicking For-EachLoop:



After Clicking For-Of Loop:



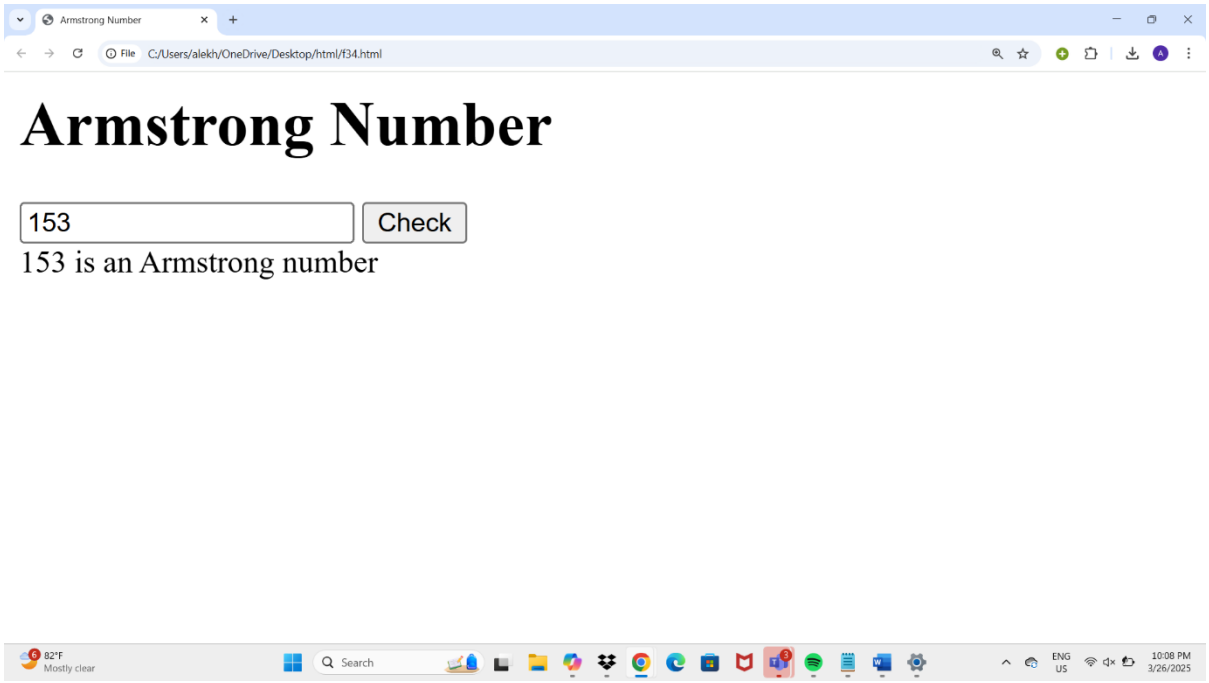
e. Develop a program to determine whether a given number is an 'ARMSTRONG NUMBER' or not. [Eg:153 is an Armstrong number, since sum of the cube of the digits is equal to the number i.e.,13 + 53+ 33 = 153]

A)

HTML Code:

```
<html
<head>
   <title>Armstrong Number</title>
</head>
<body>
   <h1>Armstrong Number</h1>
   <input id="num" type="number" placeholder="Enter a number">
   <button onclick="check ()">Check</button>
   <div id="result"></div>
   <script>
      function check () {
         let num = parseInt(document.getElementById("num").value);
         let strNum = num.toString();
         let sum = 0;
         for (let i = 0; i < strNum.length; i++) {
            let digit = parseInt(strNum[i]);
            sum += Math.pow(digit, strNum.length);
         }
         if (sum === num) {
            document.getElementById("result").innerHTML = `${num} is an
Armstrong number`;
         } else {
            document.getElementById("result").innerHTML = `${num} is not an
Armstrong number`;
         }
      }
   </script>
</body>
</html>
```

OUTPUT:

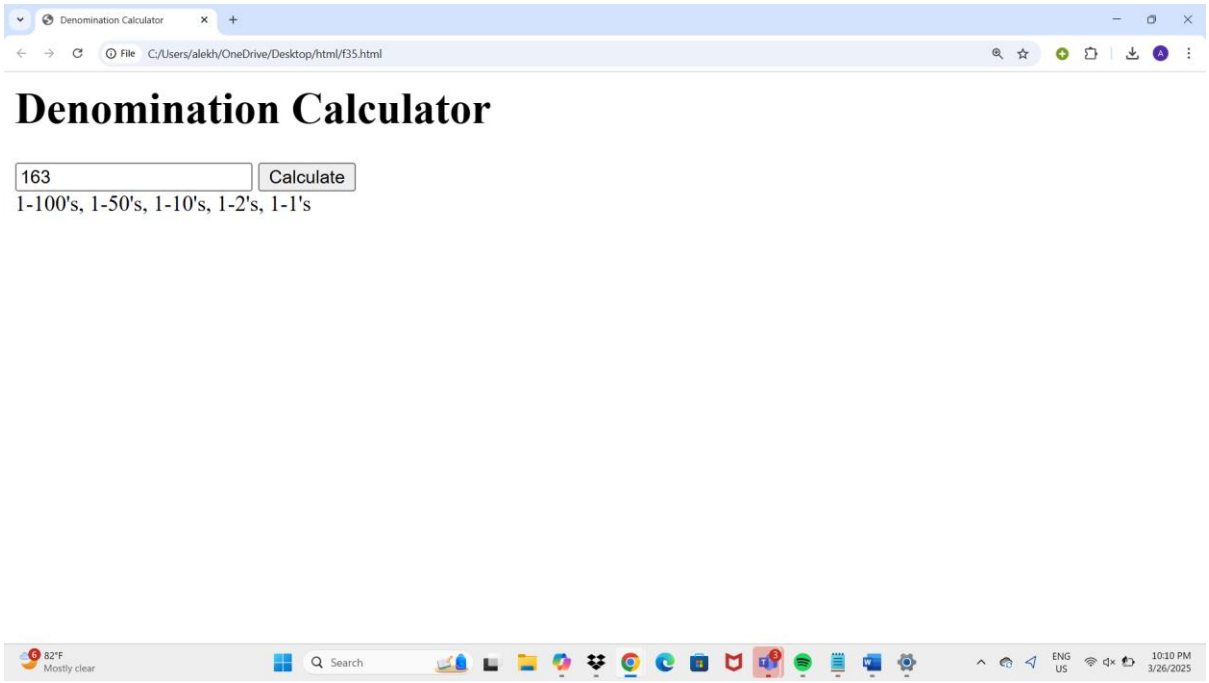


f. Write a program to display the denomination of the amount deposited in the bank in terms of 100's, 50's, 20's, 10's, 5's, 2's & 1's. (Eg: If deposited amount is Rs.163, the output should be 1-100's, 1-50's, 1- 10's, 1-2's & 1-1's)
A)

HTML Code:

```
<html>
<head>
   <title>Denomination Calculator</title>
</head>
<body>
   <h1>Denomination Calculator</h1>
   <input id="amount" type="number" placeholder="Enter amount">
   <button onclick="calculateDenomination()">Calculate</button>
   <div id="result"></div>
   <script>
      function calculateDenomination() {
         let amount = parseInt(document.getElementById("amount").value);
         let denominations = [
            { value: 100, count: 0 },
            { value: 50, count: 0 },
            { value: 20, count: 0 },
            { value: 10, count: 0 },
            { value: 5, count: 0 },
            { value: 2, count: 0 },
            { value: 1, count: 0 }
         ];
         for (let i = 0; i < denominations.length; i++) {
            denominations[i].count = Math.floor(amount / denominations[i].value);
            amount %= denominations[i].value;
         }
         let result = "";
         for (let i = 0; i < denominations.length; i++) {
            if (denominations[i].count > 0) {
               result += `${denominations[i].count}-${denominations[i].value}'s, `;
            }
         }
         document.getElementById("result").innerHTML = result.slice(0, -2);
}
   </script>
</body>
</html>
```

OUTPUT:


Denomination Calculator
File C:/Users/alekh/OneDrive/Desktop/html/f35.html
Denomination Calculator
163
Calculate
1-100's, 1-50's, 1-10's, 1-2's, 1-1's
82°F
Mostly clear
Search
ENG
US
10:10 PM
3/26/2025