a) What is the difference between a block-level and an inline-level element in HTML?

b) Write an HTML code snippet that creates a form with a mandatory text input field and a submit button.

c) Explain how <canvas> element works and provide a simple example of its usage.

d) What is the <iframe> element? How it is different from <embed> and <object>?

Here are the responses to the questions in the image:

---

a) Difference between block-level and inline-level elements in HTML:

- **Block-level elements:**
  - Always start on a new line and take up the full width available.
  - Examples: `<div>`, `<p>`, `<h1>`, `<section>`, etc.
  - Used for structuring content (e.g., paragraphs, sections).
- **Inline-level elements:**
  - Do not start on a new line and only take up as much width as necessary.
  - Examples: `<span>`, `<a>`, `<strong>`, `<em>`, etc.
  - Used for formatting text or content within a block.

---

b) HTML code snippet for a form with a mandatory text input field and a submit button:

```
<form action="/submit" method="post">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" required>
  <button type="submit">Submit</button>
</form>
```

---

c) How `<canvas>` works and example:

- The `<canvas>` element is used to draw graphics (like charts, shapes, and animations) using JavaScript.

- It provides a 2D drawing context where developers can render visual content dynamically.

Example:

```
<canvas id="myCanvas" width="200" height="100" style="border:1px solid black;"></canvas>
<script>
  const canvas = document.getElementById('myCanvas');
  const ctx = canvas.getContext('2d');
  ctx.fillStyle = 'blue';
  ctx.fillRect(50, 20, 100, 50); // Draws a blue rectangle
</script>
```

---

d) What is the `<iframe>` element? Differences from `<embed>` and `<object>`:

- **`<iframe>`:**
    - Embeds another HTML document within the current page.
    - Commonly used to embed external web pages, videos (e.g., YouTube), or ads.
- **Differences:**
    - **`<iframe>`**: Specifically for embedding HTML documents.
    - **`<embed>`**: Used for embedding external content like audio, video, or plugins (e.g., Flash).
    - **`<object>`**: A general-purpose container for embedding external resources like PDFs, Flash, or images.

Example of `<iframe>`:

```
<iframe src="https://www.example.com" width="600" height="400"
style="border:none;"></iframe>
```

a) Distinguish between "padding" and "margin" area of CSS box model.

b) What are the differences between GET and POST methods in form submitting? Give the case where we can use GET and where we can use POST methods.

c) What does HTML stand for? Write the image tag required to link from the page *links.html* to the image *contact.jpg* when the files are organized as follows:

i. both in the same folder
ii. the *contact.jpg* is located in a subfolder called *"images"*
iii. the *links.html* page is located in a subfolder called *"otherfiles"*

d) Write the differences between "id" and "class" attribute of an HTML element.

Here are the answers to the questions in the image:

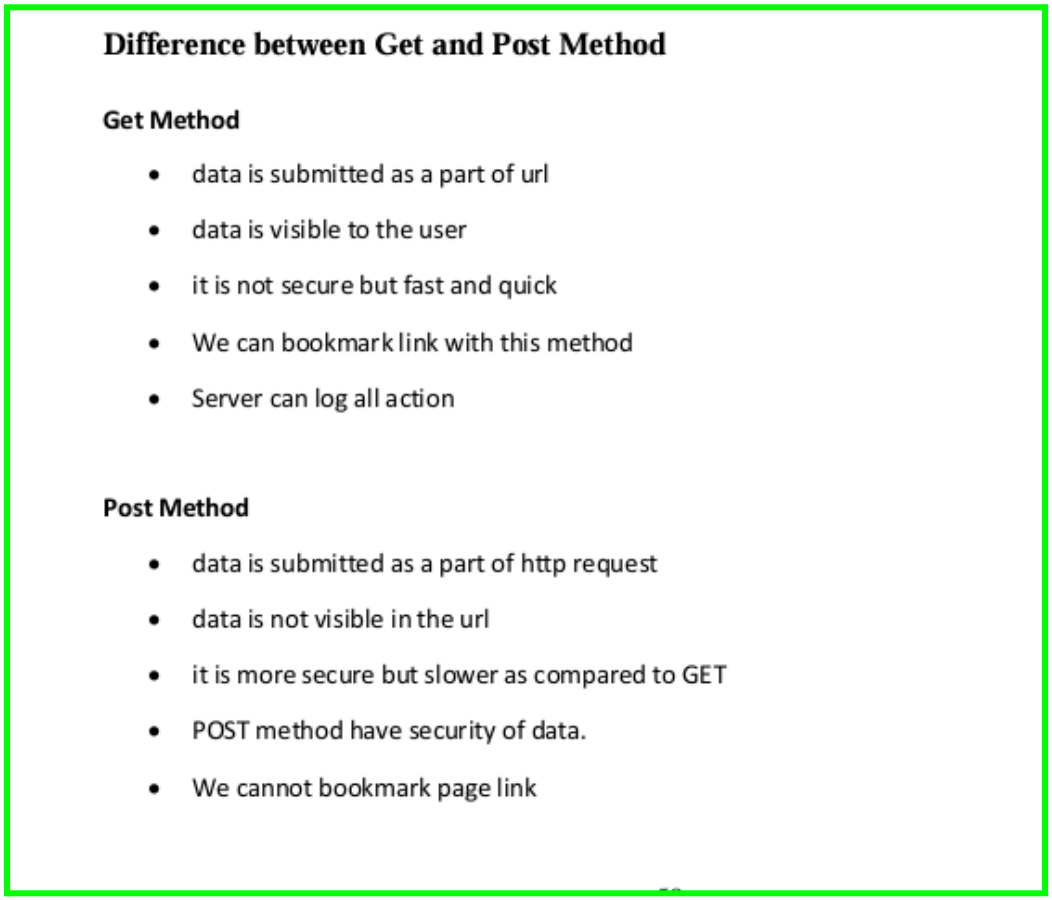
---

a) Distinction between "padding" and "margin" in the CSS box model:

- **Padding:**
  - Space between the content of an element and its border.
  - Affects the inner spacing of the element.
- **Margin:**
  - Space outside the border, separating the element from other elements.
  - Affects the outer spacing of the element.

---

b) Differences between GET and POST methods in form submission:



Difference between Get and Post Method

Get Method

- data is submitted as a part of url
- data is visible to the user
- it is not secure but fast and quick
- We can bookmark link with this method
- Server can log all action

Post Method

- data is submitted as a part of http request
- data is not visible in the url
- it is more secure but slower as compared to GET
- POST method have security of data.
- We cannot bookmark page link

- **GET:**
    - Sends data as URL parameters (query string).
    - Data is visible in the URL and has a size limit.
    - Suitable for retrieving data (e.g., search queries).

Example:

```
 <form action="/search" method="get">
  <input type="text" name="query">
  <button type="submit">Search</button>
</form>
```

- **POST:**
    - Sends data in the request body, not visible in the URL.
    - Suitable for submitting sensitive or large data (e.g., login forms).

Example:
```
 <form action="/submit" method="post">
  <input type="text" name="username">
  <button type="submit">Submit</button>
</form>
```

---

c) What HTML stands for and image linking:

- **HTML**: HyperText Markup Language.

Image tags for linking `contact.jpg` from `links.html`:

i) **Both in the same folder:**
```
<img src="contact.jpg" alt="Contact">
```

ii) **`contact.jpg` in a subfolder named "images":**
```
<img src="images/contact.jpg" alt="Contact">
```

iii) **`links.html` in "otherfiles":**
```
<img src="../contact.jpg" alt="Contact">
```

---

d) Differences between `id` and `class` attributes:

- **`id`:**
    - Uniquely identifies a single element on the page.
    - Used for JavaScript and CSS targeting specific elements.

Example:
```
 <div id="header">Header</div>
```

- **`class`:**
    - Can be assigned to multiple elements.
    - Used to group elements for styling or behavior.

Example:
```
 <div class="content">Content 1</div>
<div class="content">Content 2</div>
```

4. a) What are JavaScript Objects? Give some examples of JavaScript built-in objects. 3

b) Write a JavaScript program to evaluate the following mathematical expression: 5

$$1 + \frac{2}{2!} + \frac{3}{3!} + \cdots + \frac{n}{n!}$$

c) Create a regex for JavaScript which can validate any email id. 2

a) What are JavaScript Objects? Give some examples of JavaScript built-in objects.

JavaScript Objects are collections of key-value pairs where keys are properties or methods, and values are the associated data or functions. They allow the grouping of related data and functionality.

Some built-in JavaScript objects are:

- `Object`: The base object for all JavaScript objects.
- `Array`: Used for storing lists of items.
- `String`: Represents a sequence of characters.
- `Number`: Represents numeric values.
- `Math`: Provides mathematical constants and functions.
- `Date`: Used to handle dates and times.
- `RegExp`: Used for regular expressions.
- `Error`: Represents runtime errors.

b) Write a JavaScript program to evaluate the following mathematical expression:

```javascript
function evaluateExpression(n) {
    let sum = 1;
    let fact = 1;

    for (let i = 2; i <= n; i++) {
        fact *= i;
        sum += i / fact;
    }

    return sum;
}

let n = 5;
console.log(evaluateExpression(n));
```

c) Create a regex for JavaScript which can validate any email id.

let emailRegex = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$/;

1. i. What will be the output of the following PHP code?

```php
<?php
$i=0;
for(++$i; ++$i; ++$i){
    print $i;
    if ($i == 4)
        break;
}
?>
```

Ans: 24

The PHP for Loop

The `for` loop is used when you know how many times the script should run.

Syntax

```
for (expression1, expression2, expression3) {
  // code block
}
```

This is how it works:

- *expression1* is evaluated once
- *expression2* is evaluated before each iteration
- *expression3* is evaluated after each iteration

ii. What will be the output of the following PHP code?

```
<?php
$names = array("Sam", "Bob", "Jack");
echo $names[0]."is the brother of ".$names[1]."and
    ".$names[1].".".$brother;
?>
```

iii. What will be the output of the following PHP code?

```
<?php
$a = "clue";
$a .= "get";
echo "$a";
?>
```

ii)

Warning: Undefined variable $brother in **D:\xampp\htdocs\phpt\index.php** on line **11**

Sam is the brother of Bob and Bob.

iii) clueget

iv. What will be the output of the following PHP code?

```
<?php
$team = "arsenal";
switch ($team) {
case "manu":
     echo "I love man u";
case "arsenal":
     echo "I love arsenal";
case "manc":
     echo "I love manc";
}
?>
```

iv) I love arsenall love manc

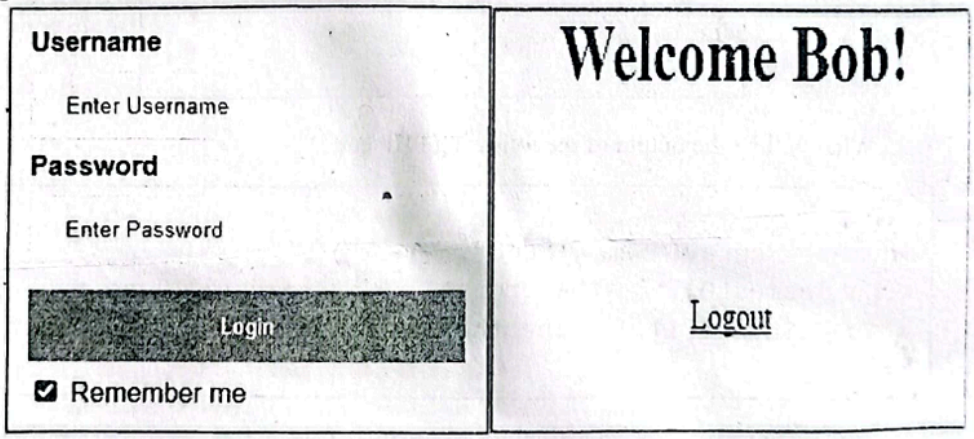
v. Find the output of the following PHP code?

```
<?php
$num = "1";
$num1 = "2";
print $num+$num1;
?>
```

v) 3

When PHP encounters the + operator with string variables, it implicitly converts these strings to integers (if they contain numeric values).

2. a) Assume, in a system **login.php, index.php, logout.php** are the three php pages. Username and password will be asked in **login.php**. **Id**, **username** and **password** are three attributes of a table "**user**" in a database "**userinfo**". The username and password should be checked for the authentication of any user in the system. If the username is found in the database, then you have to redirect to the **index.php** by setting the session variable by user id. Implement the appropriate logout by destroying session variables and associated session data. When any user checks the remember me option, set the cookies expiration time to one day. A simple scenario is given in Figure 1 for more clarification. 7



login.php *index.php*

Figure: 1

Design a proper PHP login-logout scheme with PHP **session, cookies** and the specification given above.

b) Explain DOM and CSS selectors with proper example. 3

b) DOM (Document Object Model) and CSS Selectors:

DOM:

The **Document Object Model (DOM)** represents the structure of an HTML document as a tree of objects. It allows developers to interact with the elements of a webpage, like accessing, modifying, or adding new content dynamically.

For example, using JavaScript, we can access and modify the DOM:

```
// Access an element by ID
let element = document.getElementById("username");
element.style.color = "red"; // Change the color of the element
```

CSS Selectors:

CSS Selectors are used to select the HTML elements to apply styles to. Common CSS selectors include:

- **ID Selector**: `#id` - Selects an element with a specific ID.
- **Class Selector**: `.class` - Selects all elements with a specific class.
- **Element Selector**: `element` - Selects all elements of a specific type.
- **Universal Selector**: `*` - Selects all elements.

For example:

```
/* CSS ID Selector */
#username {
    font-size: 20px;
}

/* CSS Class Selector */
.button {
    background-color: blue;
}

/* CSS Element Selector */
p {
    color: green;
}

/* Universal Selector */
* {
    margin: 0;
    padding: 0;
}
```

This allows developers to target HTML elements and style them accordingly.

Summary Table:

Feature	DOM	CSS Selectors
Purpose	Interact with the document (content/structure)	Style the document's appearance
Used With	JavaScript (for dynamic behavior)	CSS (for static styling)
Example of Operation	Changing text content, adding/removing elements	Changing font size, text color, layout
Used For	Content manipulation, event handling	Styling elements

Difference between HTML and XHTML

The key differences between HTML (HyperText Markup Language) and XHTML (Extensible HyperText Markup Language) revolve around syntax rules, flexibility, and compatibility with XML.

Summary Table:

Feature	HTML	XHTML
Syntax Rules	Less strict, flexible	Strict, follows XML rules
Case Sensitivity	Not case-sensitive	Case-sensitive (all lowercase)
Tag Closure	Optional tag closure (e.g., `<p>`)	Required tag closure (e.g., `<p></p>`)
Self-Closing Tags	Not required to close (`<img>`, ` `)	Must close (`<img />`, ` `)
DOCTYPE	Flexible DOCTYPE	Specific DOCTYPE required
Error Handling	Browsers are forgiving	Browsers reject invalid documents
Document Structure	Tolerates incorrect nesting	Requires correct nesting of elements
Content-Type Header	`text/html`	`application/xhtml+xml`

proper example. (5 marks)

In JavaScript, there are **three types of popup boxes**:

1. **Alert Box:**
   - It is used to display a message to the user.
   - The alert box only has an **OK** button.
   - Example:

```javascript
alert("This is an alert box!");
```

2. **Confirm Box:**
   - It is used to ask the user for confirmation (Yes/No).
   - It returns `true` if the user clicks **OK**, and `false` if the user clicks **Cancel**.
   - Example:

```javascript
let result = confirm("Do you want to proceed?");
if (result) {
  alert("You clicked OK!");
} else {
  alert("You clicked Cancel!");
}
```

3. **Prompt Box:**
   - It is used to ask the user for input.
   - It displays a text box along with **OK** and **Cancel** buttons. It returns the input value if **OK** is clicked or `null` if **Cancel** is clicked.
   - Example:

```javascript
let name = prompt("Enter your name:");
alert("Hello, " + name);
```

4. What is JavaScript? What is jQuery? Write down the differences between them. (4 marks)

JavaScript: JavaScript is a **programming language** used to create dynamic content on web pages. It allows developers to manipulate HTML, CSS, and handle events like clicks, input, etc.

jQuery: jQuery is a **JavaScript library** that simplifies tasks like DOM manipulation, event handling, animation, and AJAX interactions. It makes JavaScript easier to use by providing a set of pre-written functions.

Differences Between JavaScript and jQuery:

Feature	JavaScript	jQuery
Type	A programming language	A library built with JavaScript
Purpose	Used to create dynamic and interactive web pages	Simplifies JavaScript tasks like DOM manipulation and AJAX
Syntax	Requires more lines of code and complex syntax	Has simpler syntax for common tasks
Compatibility	Works across all browsers but can be more complex	Handles cross-browser compatibility automatically
Performance	Can be faster when writing optimized code	May be slower due to abstraction and extra code

Here are the answers to the questions in the image:

1. Write a function in JavaScript that will toggle the display of a HTML element. (2 marks)

You can use the following JavaScript function to toggle the display of an HTML element:

```
function toggleDisplay(elementId) {
    var element = document.getElementById(elementId);
    if (element.style.display === "none") {
        element.style.display = "block";
    } else {
```

```
        element.style.display = "none";
    }
}
```

This function checks whether the element's display property is set to `none` (hidden). If it is, the element is displayed; otherwise, it is hidden.

2. What is the difference between HTML and HTML5? What are the new features introduced in HTML5? (2+2 marks)

Difference Between HTML and HTML5:

- **HTML**: HTML (HyperText Markup Language) is the standard markup language used to create web pages and web applications. It defines the structure of web content using elements like `<div>`, `<p>`, `<img>`, etc.
- **HTML5**: HTML5 is the latest version of HTML. It includes improvements in multimedia support, new APIs, better semantics, and overall performance. It is designed to be cleaner, more modern, and provide support for mobile devices and rich web applications.

New Features in HTML5:

- **Canvas**: `<canvas>` element allows drawing of graphics, such as lines, rectangles, and images, using JavaScript.
- **Geolocation**: Provides access to a user's geographic location (latitude and longitude) via the browser.
- **New form elements**: Includes new input types like `email`, `tel`, `number`, `date`, `range`, etc. Also includes new form attributes like `placeholder` and `required`.

3. Write short notes on the following: (2+2+2 marks)

- **Canvas**:

The `<canvas>` element in HTML5 is used to draw graphics on the fly via JavaScript. It is commonly used for creating game graphics, animations, and interactive graphics. For example:

```
 <canvas id="myCanvas" width="500" height="500"></canvas>
<script>
```

```
    var canvas = document.getElementById("myCanvas");
    var ctx = canvas.getContext("2d");
    ctx.fillStyle = "red";
    ctx.fillRect(0, 0, 100, 100);
</script>
```

- **Geolocation**:

The Geolocation API provides access to the user's location. It can be used to get the user's geographical coordinates (latitude and longitude), and developers can use it for location-based services, such as maps or weather apps.

```
 if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(function(position) {
        var latitude = position.coords.latitude;
        var longitude = position.coords.longitude;
        console.log("Latitude: " + latitude + " Longitude: " + longitude);
    });
}
```

- **New Form Elements in HTML5**:
    - HTML5 introduced new form elements that improve the user experience by providing better form validation, input types, and other enhancements:
        - `email`, `tel`, `url`, `number`, `date`, `range`, etc.

For example, the `email` input type helps validate email formats automatically.

```
 <input type="email" placeholder="Enter your email">
```

4. What are the security vulnerabilities developers face when creating Web Applications? (2 marks)

Security vulnerabilities in Web Applications:

- **SQL Injection**: Attackers manipulate SQL queries by injecting malicious SQL code through form inputs, which can lead to unauthorized access to the database.

- **Cross-Site Scripting (XSS)**: Attackers inject malicious scripts into web pages, which are then executed by users' browsers. This can steal sensitive data like login credentials.
- **Cross-Site Request Forgery (CSRF)**: Attackers trick authenticated users into making unintended requests, leading to unwanted actions being performed in the web application.
- **Session Hijacking**: Attackers steal session tokens to impersonate users.

5. What is DOS attack? How to protect yourself from DOS? (3 marks)

DOS (Denial of Service) Attack:

- A DOS attack aims to make a service unavailable to users by overwhelming it with traffic. Attackers flood a server with excessive requests, causing it to crash or slow down, denying service to legitimate users.

How to Protect from DOS:

- **Firewalls and Intrusion Detection Systems**: These can detect and block malicious traffic.
- **Rate Limiting**: Limit the number of requests from a user within a specific time frame.
- **Content Delivery Networks (CDNs)**: Distribute the traffic across multiple servers, making it harder for attackers to overload the system.
- **DDoS Protection Services**: Use services like Cloudflare to protect against large-scale attacks.

6. Write short notes on the following: (3+3 marks)

- **SQL Injection**:
    - SQL Injection occurs when an attacker injects malicious SQL code into a query. This can allow attackers to view, modify, or delete data in a database. To prevent SQL Injection, developers should use parameterized queries or prepared statements.
- **XSS (Cross-Site Scripting)**:
    - XSS is a vulnerability that allows attackers to inject malicious scripts into web pages viewed by other users. These scripts can steal sensitive data, such as cookies and session tokens. To prevent XSS, developers should sanitize and escape user inputs.

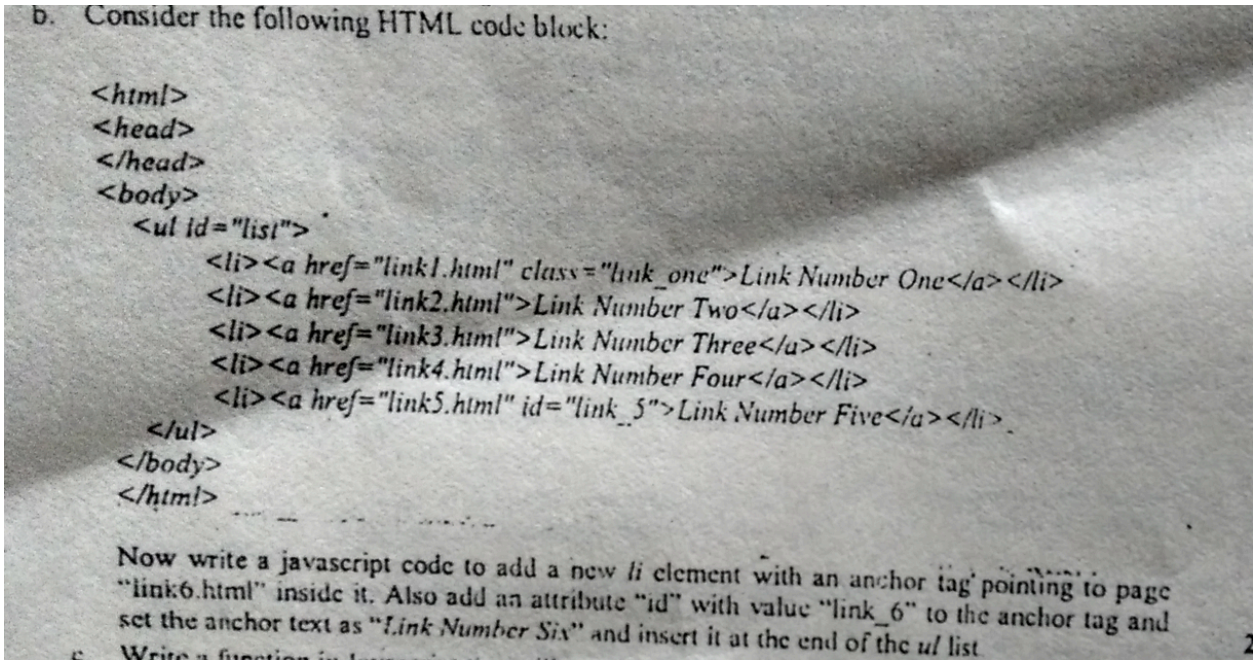
7. What is a Web Service? Why is a Web Service becoming popular nowadays? (2 marks)

Web Service:

- A **Web Service** is a software system designed to support interoperable machine-to-machine interaction over a network. Web services allow different applications to communicate with each other using standard protocols like HTTP, SOAP, REST, etc.

Why Web Services Are Becoming Popular:

- **Interoperability**: Web services allow applications built on different platforms to communicate and exchange data.
- **Scalability**: Web services can be accessed over the internet, making it easier to scale services.
- **Ease of Integration**: They simplify the process of integrating systems and sharing data between different software applications.



b. Consider the following HTML code block:

```
<html>
<head>
</head>
<body>
  <ul id="list">
      <li><a href="link1.html" class="link_one">Link Number One</a></li>
      <li><a href="link2.html">Link Number Two</a></li>
      <li><a href="link3.html">Link Number Three</a></li>
      <li><a href="link4.html">Link Number Four</a></li>
      <li><a href="link5.html" id="link_5">Link Number Five</a></li>
  </ul>
</body>
</html>
```

Now write a javascript code to add a new *li* element with an anchor tag pointing to page "link6.html" inside it. Also add an attribute "id" with value "link_6" to the anchor tag and set the anchor text as "*Link Number Six*" and insert it at the end of the *ul* list

```
// Create a new list item
var newListItem = document.createElement("li");

// Create a new anchor element
var newAnchor = document.createElement("a");
newAnchor.href = "link6.html"; // Set the link to "link6.html"
newAnchor.id = "link_6"; // Set the id to "link_6"
newAnchor.textContent = "Link Number Six"; // Set the anchor text

// Append the anchor element to the list item
newListItem.appendChild(newAnchor);

// Find the unordered list by its id and append the new list item at the end
var ulList = document.getElementById("list");
ulList.appendChild(newListItem);
```

Here are the answers to the questions based on the image you uploaded:

3. a) List the HTML elements that you need to create a table and explain how each element is used. (4 marks)

To create a table in HTML, you typically use the following HTML elements:

1. **`<table>`**: Defines the start of the table. This tag is used to wrap all table elements.
    - Example: `<table>...</table>`
2. **`<tr>`**: Defines a table row. This tag is used to create a row within the table.
    - Example: `<tr>...</tr>`
3. **`<th>`**: Defines a table header cell. It is used within a table row (`<tr>`) and contains header information.
    - Example: `<th>Header 1</th>`
4. **`<td>`**: Defines a table data (cell). It is used to create the individual cells of a table.
    - Example: `<td>Cell 1</td>`

Example of a simple table:

```
<table>
  <tr>
    <th>Header 1</th>
```

```
    <th>Header 2</th>
  </tr>
  <tr>
    <td>Row 1, Cell 1</td>
    <td>Row 1, Cell 2</td>
  </tr>
</table>
```

---

3. b) Every element has an opening and closing tag. For example, `<p>...</p>`. Identify two HTML elements that combine the opening and closing tag into one single tag. Explain the difference between the `alt` attribute and the `title` attribute. (3 marks)

HTML elements with a single self-closing tag:

1. `<img>`: Used to embed images on a page.
    - Example: `<img src="image.jpg" alt="Description of image" />`
2. `<br>`: Used to insert a line break.
    - Example: `Line 1<br>Line 2`

Difference between the `alt` and `title` attributes:

- **`alt` attribute**:
    - The `alt` attribute provides alternative text for an image if it cannot be displayed. It is important for accessibility, especially for screen readers.
    - Example: `<img src="image.jpg" alt="A beautiful sunset" />`
- **`title` attribute**:
    - The `title` attribute provides additional information about an element, usually displayed as a tooltip when the user hovers over the element.

- Example: `<a href="https://example.com" title="Click to visit the website">Visit Website</a>`

---

3. c) In HTML, how is multimedia added? List the merits and demerits of the approach. (3 marks)

How multimedia is added in HTML:

Multimedia like images, audio, and video are added using HTML tags like `<img>`, `<audio>`, and `<video>`.

1. **Images**:
    - Example: `<img src="image.jpg" alt="A scenic view" />`
2. **Audio**:

Example:

```
<audio controls>
  <source src="audio.mp3" type="audio/mp3">
  Your browser does not support the audio element.
</audio>
```

    -
3. **Video**:

Example:

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  Your browser does not support the video element.
</video>
```

    -

Merits:

- **Convenience**: HTML5 makes it easy to embed multimedia directly within the page.
- **Cross-Platform**: Multimedia elements are widely supported by modern browsers without the need for plugins like Flash.

Demerits:

- **Compatibility Issues**: Older browsers may not support certain multimedia formats like HTML5 audio/video or specific file types (e.g., `.ogg` for audio).
- **Performance**: Embedding large multimedia files directly on a page can slow down load times and affect the performance of the website.