FWT EXAM PREPRATION

CH.1 INTRODUCTION TO WEB TECHNOLOGY

1) Define Internet and WWW.

Internet:
The Internet is a global network of interconnected computers that communicate using standard protocols. It allows users to share information, access resources, and connect worldwide.

WWW (World Wide Web):
The WWW is a collection of web pages and web resources accessed through the Internet using a web browser. It uses hyperlinks and URLs to navigate between pages.

2) What is HTTP? Write its full form.

HTTP (HyperText Transfer Protocol) is a communication protocol used for transferring web pages from a web server to a web browser.

3) What is the difference between HTTP and HTTPS?

- **HTTP:** Unsecured communication; data is sent as plain text.
- **HTTPS:** Secured communication using encryption (SSL/TLS). Data is safe from attackers.

4) Define a web browser.

A **web browser** is a software application used to access, view, and browse web pages on the World Wide Web.

5) What is a URL? Give an example.

A **URL (Uniform Resource Locator)** is the address used to locate a resource on the Internet.
Example: https://www.google.com

6) List any two functions of a web client.

1. Sends HTTP requests to web servers.
2. Displays web pages received from servers.

7) Define FTP and state its use.

FTP (File Transfer Protocol) is a protocol used to upload and download files between a client and a server on the Internet.

8) Compare HTTP and HTTPS with an example.

HTTP	HTTPS
Not encrypted	Encrypted
URL starts with **http://**	URL starts with **https://**
Less secure	Highly secure

Example:
http://example.com vs https://example.com

9) Explain the structure of a URL.

A URL has these main parts:

1. **Protocol** – e.g., https
2. **Domain name** – e.g., google.com
3. **Path** – e.g., /search
4. **Parameters** – e.g., ?q=chatgpt

Example: https://www.example.com/page?id=10

10) What are the major functions of a web browser?

- Rendering and displaying web pages
- Managing bookmarks
- Handling cookies and cache
- Executing JavaScript
- Ensuring secure browsing (HTTPS support)

11) Differentiate between Internet and WWW.

- **Internet:** Physical network of computers.
- **WWW:** Collection of web pages stored on the Internet.

12) Explain the working of FTP.

FTP works by establishing a connection between an FTP client and an FTP server.

- The client sends a request to download/upload files.
- The server authenticates the user.
- Data transfer occurs through data and control channels.

13) Explain Internet, WWW, HTTP, and HTTPS in detail with examples.

Internet: A global network connecting millions of devices.
WWW: A system of interlinked hypertext documents accessed via the Internet.
HTTP: Protocol for transferring web pages.
HTTPS: Secure version of HTTP with encryption.

Example:
https://en.wikipedia.org loads via HTTPS from the WWW over the Internet.

14) What is a web browser? Explain its components and functionalities.

A web browser is software to access the web.

Components:

- **User Interface**
- **Rendering Engine**
- **JavaScript Engine**
- **Networking Module**
- **Data Storage (cache, cookies)**

Functionalities:

- Displays web pages
- Handles security
- Loads HTML, CSS, JS
- Downloads files

15) What is FTP? Describe how file transfer occurs using FTP.

FTP allows file transfer between client and server.
Transfer steps:

1. Client connects to server
2. Login/authentication

3. File upload/download
4. Server sends confirmation

16) Describe the architecture of a web client and its role in web communication.

A web client consists of:

- **User Interface**
- **Networking Module**
- **Rendering Engine**
- **Scripting Engine**

Role:

- Sends HTTP requests
- Receives and displays responses
- Handles cookies, cache
- Makes communication possible with servers

17) Discuss the difference between URL and URI with proper format.

- **URL (Locator):** Location of a resource.
  Example: https://example.com/about
- **URI (Identifier):** Can identify a resource even without giving a location.
  Example: mailto:info@example.com

18) What is a web server?

A **web server** is software or hardware that stores web pages and delivers them to clients on request.

19) List any two features of a web server.

1. Handles HTTP requests
2. Provides security and access control

20) What is access control in a web server?

Access control restricts which users or systems can access specific files or resources on the server.

21) What is server-side logging?

Server-side logging records activities such as user requests, errors, and server status for monitoring and troubleshooting.

22) Write a short note on the history of web servers.

The first web server was created by Tim Berners-Lee in 1990 (CERN).
Later, Apache became the most popular server in the early 2000s.
Modern servers include Nginx, IIS, and cloud-based services.

23) Explain any three features of a web server.

1. **Request Handling**
2. **Security Support (HTTPS)**
3. **Logging and Monitoring**

24) What is the use of logging in web servers?

Logging helps administrators track errors, performance, and user activity. It is essential for security audits and troubleshooting.

25) Define access control. How does it secure a web server?

Access control ensures only authorized users can access resources.
It improves security by preventing unauthorized access or attacks.

26) Explain the main features of a web server in detail.

- Request processing
- Content management
- Security support
- Logging
- Session handling
- Scalability features

27) Describe the history and evolution of web servers.

- 1990: First server at CERN
- 1995: Apache introduced
- 2004: Nginx for high performance
- Modern era: Cloud servers, load balancers, container-based servers

28) What is logging in a web server? How is it useful in administration?

Logging records all server activities.
Helps in monitoring performance, detecting attacks, analyzing traffic, and fixing issues.

29) Discuss access control mechanisms used in web servers.

1. **Password-based access**
2. **IP-based restrictions**
3. **Role-based access control (RBAC)**
4. **SSL certificates**

30) Explain the working of a web server with a labeled diagram.

Working:

1. Client sends HTTP request
2. Server processes request
3. Server fetches required file/page
4. Server sends back response
5. Browser displays page

31. Differentiate between HTTP and HTTPS with examples.

HTTP (HyperText Transfer Protocol):

- Used for transferring web pages over the Internet.
- The data is *not encrypted* and can be intercepted.
- Suitable for general browsing where security is not critical.
- URL starts with **http://**

HTTPS (HyperText Transfer Protocol Secure):

- Secure version of HTTP that uses **SSL/TLS encryption**.
- Protects data from hackers during transmission.
- Used for banking, payments, login pages, etc.
- URL starts with **https://**

Example:

- HTTP → http://example.com
- HTTPS → https://example.com (secure)

32. Explain URL and its components with example.

A **URL (Uniform Resource Locator)** is the address used to access a resource on the Internet.

Components of a URL:

1. **Protocol** – Defines communication type (e.g., https, ftp)
2. **Domain Name** – Name of the website (e.g., google.com)
3. **Port Number** (optional) – e.g., :80, :443
4. **Path** – Specific location/resource on the server (e.g., /products)
5. **Query Parameters** (optional) – Extra information (e.g., ?id=10)

Example:

https://www.amazon.com/electronics?category=mobile

- Protocol → https
- Domain → amazon.com
- Path → /electronics
- Query → ?category=mobile

33. What is the difference between Internet and World Wide Web (WWW)?

Internet:

- A global network of interconnected computers.
- Provides communication services like email, file sharing, video calls, etc.
- Hardware + network infrastructure.

WWW:

- A collection of web pages, websites, and multimedia content accessible using the Internet.
- Uses HTTP/HTTPS protocols.
- Part of the Internet, not the whole Internet.

In short:
Internet = physical network
WWW = information on that network

34. What are the functions of a web browser?

1. Fetches web pages from web servers using HTTP/HTTPS
2. Renders HTML, CSS, and JavaScript to display webpages
3. Provides user interface (tabs, back/forward, bookmarks, history)
4. Manages cookies, cache, and session data
5. Ensures security using SSL certificates
6. Allows downloading files and saving webpages

35. Explain FTP and how it works for file transfer.

FTP (File Transfer Protocol):
A protocol used to upload/download files between a client and a server over a network.

How FTP works:

1. **Connection Establishment:**
   FTP client connects to the FTP server using the server address.
2. **Authentication:**
   User logs in using a username and password (or anonymous login).
3. **File Transfer Mode:**
   - ASCII mode → for text files
   - Binary mode → for images, videos, software
4. **Transfer Process:**
   The client sends commands and the server sends back files using separate control and data channels.
5. **Completion:**
   The server confirms successful upload/download.

36. What is a web server? Explain its working.

A **web server** is software or hardware that stores web pages and delivers them to clients on request.

Working of a Web Server:

1. **Client Request:**
   Browser sends an HTTP/HTTPS request to the server.
2. **Processing:**
   The server checks the request, locates the requested file/page, or executes scripts if needed.

3. **Response:**
   The server sends back an HTML page or data along with a status code (e.g., 200 OK).
4. **Rendering:**
   Browser displays the content to the user.

37. Write a short note on the features of a web server.

1. **Request Handling:**
   Processes client HTTP/HTTPS requests efficiently.
2. **Security Support:**
   Supports SSL/TLS for secure communication.
3. **Logging:**
   Records user activity, errors, and server performance.
4. **Load Handling:**
   Can manage multiple users at the same time.
5. **Content Management:**
   Stores and serves web pages, images, scripts, and files.

38. Explain the concept of server-side logging. Why is it important?

Server-side logging refers to recording all significant activities performed on the server, such as requests, responses, errors, and user actions.

Importance of server-side logging:

- Helps detect errors and troubleshoot issues
- Monitors server performance
- Helps identify security threats and unauthorized access
- Useful for auditing and analyzing website traffic
- Ensures smooth administration of the server

39. What is access control in a web server? How does it enhance security?

Access Control is a security mechanism used to restrict who can access certain files, directories, or resources on a web server.

How it enhances security:

- Prevents unauthorized users from accessing sensitive data
- Protects admin panels and confidential files
- Ensures only authenticated users can access specific areas
- Helps stop hacking attempts and misuse

Types:

- Password-based access
- IP-based restrictions
- Role-Based Access Control (RBAC)

40. Explain the history and evolution of web servers.

- **1990:**
  The first web server (CERN httpd) was created by Tim Berners-Lee at CERN.
- **Mid-1990s:**
  Apache HTTP Server released → became the most popular server due to open-source support.
- **2000s:**
  Microsoft IIS (Internet Information Services) gained popularity on Windows platforms.
- **2004:**
  Nginx introduced → optimized for speed and handling high-traffic websites.
- **Present:**
  Modern servers include cloud-based systems like AWS, Google Cloud, and containerized servers using Docker & Kubernetes.

Evolution trend:
Basic text servers → dynamic content → secure HTTPS → scalable cloud servers

CH.2 HTML, XHTML & HTML5

1. Explain the structure of an HTML document with a neat diagram.

An HTML document follows a proper structure that helps the browser understand the content.

Structure:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>

  <body>
    <h1>Main Content</h1>
    <p>This is a paragraph.</p>
  </body>
</html>
```

Explanation of Sections:

- **<!DOCTYPE html>** → Declares HTML5 document
- **<html>** → Root element
- **<head>** → Contains metadata (title, meta tags, styles)
- **<title>** → Name of the webpage
- **<body>** → Displays all visible content

Neat Diagram (ASCII):

```
HTML Document
 ├── <!DOCTYPE html>
 └── <html>
      ├── <head>
      │    ├── <title>
      │    └── Metadata
      └── <body>
           ├── Headings
           ├── Paragraphs
           ├── Images
           └── Links
```

2. What are lists in HTML? Explain ordered, unordered, and definition lists with examples.

HTML lists are used to display items in a structured way.

1. Ordered List (<ol>) – numbered list

```
<ol>
  <li>Apple</li>
  <li>Banana</li>
  <li>Mango</li>
</ol>
```

2. Unordered List (<ul>) – bullet list

```
<ul>
  <li>Pen</li>
  <li>Pencil</li>
  <li>Book</li>
</ul>
```

3. Definition List (<dl>) – term & definition

```
<dl>
  <dt>HTML</dt>
  <dd>HyperText Markup Language</dd>
</dl>
```

3. Explain the navigation structure of a website. Write HTML code to open a link in a new tab/window.

Navigation structure represents how users move from one page to another.
It usually contains menus like:

- Home
- About
- Services
- Contact

A navigation menu is created using **links (<a>)**.

HTML code to open a link in a new tab:

```
<a href="https://www.google.com" target="_blank">Open Google</a>
```

target="_blank" opens link in a new tab/window.

4. What is an HTML form? Explain <form> **attributes and design a student registration form.**

HTML Form:

A form collects user input and sends it to a server.

Important <form> Attributes:

- **action** → URL to send form data
- **method** → GET or POST
- **enctype** → Type of data encoding
- **name** → Name of the form
- **target** → Where to display result

Student Registration Form:

```
<form action="submit.php" method="post">
  <h3>Student Registration</h3>

  Name: <input type="text" name="name"><br><br>
  Email: <input type="email" name="email"><br><br>
  Age: <input type="number" name="age"><br><br>

  Gender:
  <input type="radio" name="gender" value="Male"> Male
  <input type="radio" name="gender" value="Female"> Female<br><br>

  Course:
  <select name="course">
    <option>BCA</option>
    <option>BBA</option>
  </select><br><br>

  <input type="submit" value="Register">
</form>
```

5. Write HTML code to create a login page with username, password, and submit button.

```
<form>
  <h2>Login Page</h2>

  Username:
  <input type="text" name="username"><br><br>

  Password:
  <input type="password" name="password"><br><br>

  <input type="submit" value="Login">
</form>
```

6. Explain HTML tables in detail. Write code for a timetable using rowspan and colspan.

HTML Table Concepts:

- **<table>** → table container
- **<tr>** → table row
- **<td>** → table data
- **<th>** → table heading
- **rowspan** → merge cells vertically
- **colspan** → merge cells horizontally

Time-Table Example:

```
<table border="1">
  <tr>
    <th colspan="5">Time Table</th>
  </tr>

  <tr>
    <th>Day</th>
    <th>9-10</th>
    <th>10-11</th>
    <th>11-12</th>
    <th>12-1</th>
  </tr>

  <tr>
```

```
    <td rowspan="2">Monday</td>
    <td>Math</td>
    <td>English</td>
    <td rowspan="2">Break</td>
    <td>Science</td>
  </tr>

  <tr>
    <td>Computer</td>
    <td>Sports</td>
    <td>Arts</td>
  </tr>
</table>
```

7. Differentiate between HTML, XHTML, and HTML5 with examples.

HTML

- Flexible, not strict
- Example: Tags may be unclosed
- <p> Hello

XHTML

- Stricter, XML-based
- All tags must be closed
- <br />, <img src="a.jpg" />

HTML5

- Modern version with new features
- Audio/video support, semantic tags
- <video controls> … </video>

8. Explain the new features of HTML5 with examples.

1. **Audio & Video support**
   <audio controls src="song.mp3"></audio>
2. **Canvas for graphics**
3. **Semantic tags** → <header>, <nav>, <footer>
4. **New form input types** → email, date, range
5. **Local Storage & Session Storage**
6. **Geolocation API**

9. What are semantic elements in HTML5? Explain with suitable examples.

Semantic elements describe meaning of content.

Examples:

- **<header>** → Top section
- **<nav>** → Navigation menu
- **<article>** → Independent article
- **<section>** → Logical section
- **<footer>** → Bottom section

Example Code:

```
<header>My Website</header>
<nav>Home | About | Contact</nav>
<section>
  <article>Article content here</article>
</section>
```

10. Write HTML code for a feedback form with text, radio buttons, checkboxes, and reset button.

```
<form>
  <h3>Feedback Form</h3>

  Name:
  <input type="text" name="name"><br><br>

  Rate Us:
  <input type="radio" name="rate" value="Good"> Good
  <input type="radio" name="rate" value="Average"> Average
  <input type="radio" name="rate" value="Bad"> Bad<br><br>

  Services Used:<br>
  <input type="checkbox" name="service" value="Support"> Support<br>
  <input type="checkbox" name="service" value="Delivery"> Delivery<br>
  <input type="checkbox" name="service" value="Website"> Website<br><br>

  <input type="reset" value="Reset Form">
  <input type="submit" value="Submit">
</form>
```

11. Explain Block-level and Inline elements in XHTML with examples.

Block-level Elements

- Start on a **new line**
- Occupy the **full width** available
- Can contain inline elements
- Examples:
  <div>, <p>, <h1>–<h6>, <ul>, <table>

Example:

```
<p>This is a block element.</p>
```

Inline Elements

- Do **not** start on a new line
- Occupy only the **required width**
- Usually used inside block elements
- Examples:
  <span>, <a>, <img>, <strong>, <small>

Example:

```
<p>This is <span>inline text</span> inside a paragraph.</p>
```

12. Write HTML5 code to embed an audio and a video file with controls.

```
<h3>Audio Example</h3>
<audio controls>
  <source src="song.mp3" type="audio/mpeg">
</audio>

<h3>Video Example</h3>
<video controls width="400">
  <source src="movie.mp4" type="video/mp4">
</video>
```

13. Compare HTML4 and HTML5 with features and examples.

HTML4

- Older version
- No built-in multimedia support

- Depends on external plugins like Flash
- No semantic elements

Example:
<div id="header">

HTML5

- New modern version
- Supports **audio**, **video**, **canvas**
- Semantic tags → <header> <section> <footer>
- Better form controls

Example:
<header>Website Title</header>

Major Differences:

HTML4	HTML5
No audio/video tags	Built-in multimedia support
No semantic tags	Semantic tags included
Strict Doctype	Simple DOCTYPE
Needs Flash	Works without Flash

14. What are deprecated tags in HTML5? List and explain with alternatives.

Deprecated tags are **removed or outdated tags** that should not be used in HTML5.

Deprecated Tags & Alternatives

Deprecated Tag	Meaning	HTML5 Alternative
<center>	Center text	text-align:center; (CSS)
<font>	Change font style	CSS font-family, color
<big> / <small>	Font size	CSS font-size

Deprecated Tag	Meaning	HTML5 Alternative
<frame>, <frameset>	Frames layout	<iframe> or CSS layout
<u>	Underline	CSS text-decoration: underline;

15. Explain the use of <meta> **tags (charset, description, keywords, refresh). Give examples.**

<meta> tags provide metadata to the browser.

1. charset

Defines character encoding.

<meta charset="UTF-8">

2. description

Describes webpage content for search engines.

<meta name="description" content="This is my website.">

3. keywords

Keywords for SEO.

<meta name="keywords" content="HTML, CSS, Web Development">

4. refresh

Reloads page automatically.

<meta http-equiv="refresh" content="5">

16. Explain frames in HTML with example. Why are they removed in HTML5?

Frames in HTML

Frames divide the browser window into multiple sections.

Example:

```
<frameset cols="50%,50%">
  <frame src="left.html">
  <frame src="right.html">
</frameset>
```

Why Frames Removed in HTML5?

- Not mobile-friendly
- Not good for SEO
- Difficult navigation
- Complex usability
- Security issues

HTML5 uses **CSS layouts**, **iframe**, and **Flexbox/Grid** instead.

17. Write HTML code for a navigation menu using anchor and list tags.

```
<ul>
  <li><a href="home.html">Home</a></li>
  <li><a href="about.html">About</a></li>
  <li><a href="services.html">Services</a></li>
  <li><a href="contact.html">Contact</a></li>
</ul>
```

18. Explain difference between <div> **and** <span> **with examples.**

<div>

- Block-level element
- Used for large sections
- Starts on a new line

Example:

```
<div>
  <h2>Content Section</h2>
</div>
```

<span>

- Inline element
- Used for small text formatting
- Does not start on a new line

Example:

```
<p>This is <span style="color:red;">highlighted</span> text.</p>
```

19. Discuss the role of HTML Validator. Why is it important in XHTML?

HTML Validator

A tool that checks HTML code for:

- Errors
- Invalid tags
- Missing closing tags
- Syntax issues

Importance in XHTML

- XHTML is strict (XML-based)
- Requires proper nesting and closing of tags
- Validator ensures:
  - Cleaner code
  - Browser compatibility
  - Better SEO
  - Fewer runtime errors

20. Write HTML code for a resume web page (with headings, lists, table, links).

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

<body>
  <h1>Atul Pal</h1>
  <h2>Contact</h2>
  <p>Email: atul@example.com</p>
  <a href="https://linkedin.com">LinkedIn Profile</a>

  <h2>Skills</h2>
  <ul>
    <li>HTML</li>
    <li>CSS</li>
    <li>C Programming</li>
  </ul>
  <h2>Education</h2>
  <table border="1">
    <tr>
      <th>Year</th>
      <th>Course</th>
      <th>Institute</th>
    </tr>
    <tr>
      <td>2024</td>
      <td>BCA</td>
      <td>GTU</td>
    </tr>
  </table>
</body>
</html>
```

21. Define HTML. List its advantages and limitations.

HTML (HyperText Markup Language)

HTML is a markup language used to create and display web pages. It uses tags to define headings, paragraphs, images, tables, forms, etc.

Advantages of HTML

1. Easy to learn and use
2. Supported by all browsers
3. Free and platform-independent
4. Integrates with CSS & JavaScript
5. Lightweight and fast to load

Limitations of HTML

1. Not secure (only a structure, no logic)
2. Cannot create dynamic pages without CSS/JS
3. Limited styling options (depends on CSS)
4. Cannot interact with databases directly
5. Difficult to maintain large HTML files

22. Explain the difference between absolute and relative hyperlinks with examples.

1. Absolute Hyperlink

- Contains complete URL
- Points to a webpage anywhere on the Internet
  Example:

```
<a href="https://www.google.com">Google</a>
```

2. Relative Hyperlink

- Points to a page within the same website
- Uses only file path, not full URL
  Example:

```
<a href="about.html">About Us</a>
```

23. Write HTML code to display 3 images in a single row.

```
<img src="img1.jpg" width="200">
<img src="img2.jpg" width="200">
<img src="img3.jpg" width="200">
```

(All <img> tags are inline elements, so they appear in one row.)

24. Write HTML code to create a list of subjects using <ol> **and** <ul>**.**

```
<h3>Ordered List</h3>
<ol>
  <li>Mathematics</li>
  <li>Physics</li>
  <li>Chemistry</li>
</ol>

<h3>Unordered List</h3>
<ul>
  <li>English</li>
  <li>Computer Science</li>
  <li>Biology</li>
</ul>
```

25. Explain the use of <head> **tag and its elements (**<title>**,** <meta>**,** <link>**).**

The **<head>** section contains metadata and information about the webpage.

1. <title> tag

- Displays the title on the browser tab

```
<title>My Webpage</title>
```

2. <meta> tag

- Provides metadata like keywords, description, charset

```
<meta charset="UTF-8">
<meta name="description" content="HTML Tutorial">
```

3. <link> tag

- Links external files (CSS, icons)

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

26. Write HTML code to display 5 headings and 3 paragraphs.

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>

<p>This is paragraph one.</p>
<p>This is paragraph two.</p>
<p>This is paragraph three.</p>
```

27. Explain the use of <br> **and** <hr> **with examples.**

1. <br> – Line Break

- Moves text to next line
- No closing tag
  Example:

```
Hello<br>World
```

2. <hr> – Horizontal Rule

- Creates a horizontal line
- Used to separate sections

  Example:

```
<h2>Title</h2>
<hr>
<p>Paragraph below line.</p>
```

28. Difference between HTML elements and attributes with examples.

HTML Elements

- Consist of start tag, content, and end tag

  Example:

```
<p>This is a paragraph.</p>
```

HTML Attributes

- Provide extra information inside the tag

  Example:

```
<img src="car.jpg" width="200">
```

Difference Table

HTML Elements	HTML Attributes
Define content	Add extra info
Have opening & closing tags	Exist inside tags
Example: <h1>	Example: src="image.jpg"

29. Write HTML code to create a form with text, password, and submit button.

```
<form>
  Username: <input type="text" name="user"><br><br>
  Password: <input type="password" name="pass"><br><br>
  <input type="submit" value="Login">
</form>
```

30. What is the difference between inline CSS and internal CSS in HTML?

Inline CSS

- Written inside the HTML tag
- Affects only one element
  Example:

```
<p style="color:red;">Hello</p>
```

Internal CSS

- Written inside <style> tag in <head>
- Affects entire page
  Example:

```
<head>
<style>
p { color: red; }
</style>
</head>
```

Difference Table

Inline CSS	Internal CSS
Applied to a single element	Applied to multiple elements
High priority	Medium priority
Hard to maintain	Easier to maintain

31. Write HTML code to create a hyperlink to another page.

```
<a href="about.html">Go to About Page</a>
```

32. Explain the difference between HTML and XHTML (rules of XHTML).

HTML

- Flexible
- Not strict
- Tags can be unclosed
- Case-insensitive

XHTML

- Stricter and XML-based
- All tags **must** be closed
- Lowercase tags only
- Attributes must have quotes
- Proper nesting required

Example Difference

HTML:

```
<br>
<img src="pic.jpg">
```

XHTML:

```
<br />
<img src="pic.jpg" alt="image" />
```

Rules of XHTML

1. Tags must be in lowercase
2. All tags must be closed
3. Attributes must be in quotes
4. Proper nesting required
5. Must include DOCTYPE for XHTML

33. What are input types in HTML5 forms? List any five with examples.

HTML5 introduced new input types:

Types with Examples:

1. **email**

```
<input type="email" name="mail">
```

2. **date**

```
<input type="date" name="dob">
```

3. **range**

```
<input type="range" min="0" max="10">
```

4. **number**

```
<input type="number" min="1" max="10">
```

5. **color**

```
<input type="color">
```

Other types: url, tel, search, time, datetime-local, week, month.

34. Write HTML code to create a table with a caption.

```
<table border="1">
  <caption>Student Marks</caption>

  <tr>
    <th>Name</th>
    <th>Marks</th>
  </tr>

  <tr>
    <td>Atul</td>
    <td>95</td>
  </tr>

  <tr>
    <td>Riya</td>
    <td>89</td>
  </tr>
</table>
```

35. Explain the use of <iframe> **with an example.**

<iframe>

- Embeds another webpage inside the current page
- Used for maps, ads, videos, external content

Example:

```
<iframe src="https://www.wikipedia.org" width="500" height="300"></iframe>
```

36. What is the difference between semantic and non-semantic tags in HTML5?

Semantic Tags

- Have meaningful names
- Clearly describe content
- Improve SEO and accessibility

Examples:
<header>, <footer>, <section>, <article>, <nav>

Non-Semantic Tags

- Do not describe meaning
- Only used for layout
- Less clear for machines

Examples:
<div>, <span>

37. List any five block-level elements and any five inline elements.

Block-level Elements

1. <div>
2. <p>
3. <h1>
4. <ul>
5. <table>

Inline Elements

1. <span>
2. <a>
3. <img>
4. <strong>
5. <em>

38. Write HTML code to display a YouTube video using <iframe>**.**

```
<iframe
  width="560"
  height="315"
  src="https://www.youtube.com/embed/VIDEO_ID"
  frameborder="0"
  allowfullscreen>
</iframe>
```

(Replace VIDEO_ID with actual YouTube video ID.)

39. Explain the purpose of DOCTYPE declaration in HTML/XHTML.

Purpose of DOCTYPE:

- Tells the browser which version of HTML/XHTML the document uses
- Ensures **standards mode**
- Prevents rendering issues
- Helps browser interpret tags correctly

HTML5 DOCTYPE:

```
<!DOCTYPE html>
```

XHTML DOCTYPE:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

40. List differences between HTML and DHTML.

HTML

- Static
- Cannot change content dynamically
- Only markup language
- No animations or interactions

DHTML (Dynamic HTML)

- Combines HTML + CSS + JavaScript
- Supports animation & dynamic effects
- Content can change without reloading
- Used for menus, slideshows, effects

Table:

HTML	DHTML
Static pages	Dynamic pages
No scripting	Uses JavaScript
Simple layout	Interactive layout
No animations	Supports animations

41. Explain the basic structure of an HTML document with an example.

An HTML document follows a fixed structure so the browser can properly display the page.

✔ **Structure:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page</title>
  </head>

  <body>
    <h1>Welcome</h1>
    <p>This is my webpage.</p>
  </body>
</html>
```

✔ **Explanation:**

- **<!DOCTYPE html>** → Declares HTML5 document
- **<html>** → Root element
- **<head>** → Contains metadata
- **<title>** → Title shown in browser tab
- **<body>** → All visible content appears here

42. What are lists in HTML? Explain ordered, unordered, and definition lists with examples.

HTML lists display items in a structured format.

✔ 1. Ordered List (Numbered)

```
<ol>
  <li>Apple</li>
  <li>Banana</li>
</ol>
```

✔ 2. Unordered List (Bulleted)

```
<ul>
  <li>Pen</li>
  <li>Pencil</li>
</ul>
```

✔ 3. Definition List

```
<dl>
  <dt>HTML</dt>
  <dd>HyperText Markup Language</dd>
</dl>
```

43. Differentiate between HTML and XHTML with suitable examples.

HTML	XHTML
Not strict	Very strict
Tags may remain unclosed	All tags must be closed
Attributes may not require quotes	All attributes must use quotes
Case-insensitive	Lowercase tags required

✔ Examples:

HTML:

```
<br>
```

<img src="pic.jpg">

XHTML:

<br />
<img src="pic.jpg" alt="image" />

44. Explain Block-level and Inline elements with examples.

✔ **Block-level Elements**

- Start on a new line
- Take full width
- Example: <div>, <p>, <h1>

<p>This is a block element.</p>

✔ **Inline Elements**

- Do not start on a new line
- Take only required width
- Example: <span>, <a>, <img>

<span>This is inline.</span>

45. What is a form in HTML? Explain different form elements with an example.

✔ **HTML Form:**

Used to collect user input.

✔ **Common Form Elements:**

- **<input type="text">** → Text box
- **<input type="radio">** → Radio button
- **<input type="checkbox">** → Checkbox
- **<textarea>** → Multi-line input
- **<select>** → Dropdown
- **<input type="submit">** → Submit button

✔ **Example:**

<form>

```
  Name: <input type="text"><br>
  Gender: <input type="radio">Male <input type="radio">Female<br>
  Hobbies: <input type="checkbox">Music<br>
  <input type="submit">
</form>
```

46. Explain HTML tables with an example showing <thead>, <tbody>, <tfoot>.

```
<table border="1">
  <thead>
    <tr><th>Name</th><th>Marks</th></tr>
  </thead>

  <tbody>
    <tr><td>Atul</td><td>90</td></tr>
    <tr><td>Riya</td><td>85</td></tr>
  </tbody>

  <tfoot>
    <tr><td>Total Students</td><td>2</td></tr>
  </tfoot>
</table>
```

47. Discuss the new features of HTML5 in detail.

1. **Audio & Video Support**
2. **Canvas** for graphics
3. **Semantic tags** → <header> <article> <footer>
4. **New input types** → email, date, range
5. **LocalStorage & SessionStorage**
6. **Geolocation API**
7. **Responsive design support**

48. Explain semantic elements in HTML5 with examples.

✔ **Examples:**

- **<header>** – Top section
- **<article>** – Independent article
- **<section>** – Group of related content
- **<footer>** – Bottom section

✔ **Example Code:**

```
<header>My Website</header>
<section>
  <article>News Article</article>
</section>
<footer>Copyright 2025</footer>
```

49. Write an HTML code to create a student registration form.

```
<form>
  Name: <input type="text"><br><br>

  Gender:
  <input type="radio" name="g">Male
  <input type="radio" name="g">Female<br><br>

  Hobbies:
  <input type="checkbox">Music
  <input type="checkbox">Sports<br><br>

  <input type="submit" value="Register">
</form>
```

50. Compare HTML4, XHTML, and HTML5.

HTML4	**XHTML**	**HTML5**
Old version	Stricter XML version	Modern version
No multimedia	Strict rules	Supports audio/video
No semantic tags	Must close tags	Semantic tags
Complex DOCTYPE	Long DOCTYPE	Simple DOCTYPE

51. Explain the use of <meta> and <title> tags.

✔ **<title>**

- Displays page title in browser

```
<title>My Page</title>
```

✔ **<meta>**

Provides metadata:

```
<meta charset="UTF-8">
<meta name="keywords" content="HTML, Web">
<meta name="description" content="Tutorial">
```

52. Differentiate between absolute and relative links with examples.

✔ **Absolute Link**

Full URL

```
<a href="https://google.com">Google</a>
```

✔ **Relative Link**

Link within same website

```
<a href="about.html">About</a>
```

53. List any five tags used in HTML with their purpose.

Tag	Purpose
<p>	Paragraph
<h1>	Heading
<img>	Insert image
<a>	Hyperlink
<table>	Create table

54. Write HTML code to create a table with 3 rows and 3 columns.

```
<table border="1">
  <tr><td>A</td><td>B</td><td>C</td></tr>
  <tr><td>D</td><td>E</td><td<F</td></tr>
  <tr><td>G</td><td>H</td><td>I</td></tr>
</table>
```

55. Explain the use of frames in HTML. Why are they deprecated in HTML5?

✔ Use of Frames:

Frames divide the window into multiple sections:

```
<frameset>
  <frame src="left.html">
  <frame src="right.html">
</frameset>
```

✔ Deprecated Because:

- Not mobile friendly
- Poor SEO
- Complicated navigation
- Security risks
- HTML5 uses CSS layout instead

56. What are HTML attributes? Explain with two examples.

Attributes give extra information to tags.

✔ Examples:

```
<img src="image.jpg" width="200">
<a href="home.html">Home</a>
```

57. Write HTML5 code to insert audio and video.

```
<audio controls>
  <source src="music.mp3" type="audio/mpeg">
</audio>

<video controls width="300">
  <source src="video.mp4" type="video/mp4">
</video>
```

58. Explain the difference between <div> and <span>.

<div>	**<span>**
Block element	Inline element
Takes full width	Takes only required width
Used for large sections	Used for small text styling

59. Write HTML code to display three headings and two paragraphs.

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>

<p>This is paragraph one.</p>
<p>This is paragraph two.</p>
```

60. List any five deprecated tags in HTML5.

1. <center>
2. <font>
3. <big>
4. <u>
5. <frameset>

CHAPTER 3

1. What do you mean by CSS declaration?

A **CSS declaration** defines how an HTML element should be styled.
It has **property : value** format.

Example:

```
color: red;
font-size: 20px;
```

2. Write the types of CSS with examples.

✔ **1. Inline CSS**

Written inside the HTML tag.

```
<p style="color: blue;">Hello</p>
```

✔ **2. Internal CSS**

Written inside <style> tag in <head>.

```
<style>
p { color: green; }
</style>
```

✔ **3. External CSS**

Written in a separate .css file.

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

3. What is the use of the “link” tag in HTML?

<link> tag is used to connect an external CSS file to an HTML document.

Example:

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

4. Define class selector and ID selector with examples.

✔ **Class Selector**

Used to style multiple elements.

```
.myClass {
  color: red;
}
```

HTML:

```
<p class="myClass">Text</p>
```

✔ **ID Selector**

Used for a **unique** element.

```
#myId {
  font-size: 20px;
}
```

HTML:

```
<p id="myId">Content</p>
```

5. What is the difference between internal and external CSS?

Internal CSS	External CSS
Written inside <style> tag	Written in .css file
Affects only that page	Can style entire website
Harder to maintain	Easier to maintain
Slows page if too large	Improves performance

6. Write the syntax to apply CSS to a paragraph element.

```
p {
  color: blue;
  font-size: 18px;
}
```

7. What is the purpose of the box model in CSS?

The **CSS box model** describes how every HTML element appears as a box.
It consists of:

- Content
- Padding
- Border
- Margin

It helps control spacing, layout, and alignment.

8. Define margin and padding in CSS.

✔ **Margin**

- Space **outside** the border
- Controls the gap between elements

✔ **Padding**

- Space **inside** the border
- Controls the space around content

Example:

```
margin: 20px;
padding: 10px;
```

9. How does the “float” property work in CSS?

float is used to position elements **left or right**, allowing text or images to wrap around them.

Example:

```
img {
  float: left;
}
```

10. What is the use of the “clear” property?

clear prevents elements from wrapping around floated elements.

Example:

```
div {
  clear: both;
}
```

11. Define “z-index” and write its purpose.

z-index controls the **stacking order** of overlapping elements.

Higher value = element appears on top.

Example:

```
div {
  z-index: 10;
  position: absolute;
}
```

12. How can CSS improve the appearance of a website?

CSS improves a website by:

- Adding colors, fonts, spacing
- Creating layouts
- Enhancing visual appeal
- Making pages responsive
- Maintaining consistency
- Reducing HTML code clutter

13. What is the role of the "default-router" property in network configuration?

→ **Not related to CSS. Ignore.**

14. How can you assign different padding values for top, right, bottom, and left?

Using **padding shorthand:**

```
padding: 10px 20px 30px 40px;
```

This means:

- Top: 10px
- Right: 20px
- Bottom: 30px
- Left: 40px

15. Mention two benefits of using external CSS.

1. Same CSS file can style **multiple webpages**
2. Website becomes lightweight and easier to maintain

16. Write the command to link an external CSS file.

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

17. Explain how CSS helps in creating responsive web pages.

CSS supports:

- **Media queries**
- **Flexible layouts**
- **Percentage-based width**
- **Responsive images**

Example (media query):

```
@media (max-width:600px) {
  div { width: 100%; }
}
```

This makes websites adapt to mobile, tablet, and desktop screens.

18. What are pseudo-classes in CSS?

Pseudo-classes define a special state of an element.

Examples:

```
a:hover   → when mouse is over
a:visited → when link is visited
input:focus → when field is active
```

19. How can CSS be used to style lists or links?

✔ **Styling Lists:**

```
ul {
  list-style-type: square;
}
```

✔ **Styling Links:**

```
a {
  color: red;
  text-decoration: none;
}
a:hover {
  color: blue;
}
```

20. Explain how CSS helps in maintaining consistency across a website.

- A single external CSS file can style all pages
- Same colors, fonts, layout everywhere
- Makes website look professional
- Easy to update — change once, apply everywhere
- Reduces errors and mismatched designs

21. What is CSS? Explain its features and importance.

✔ **CSS (Cascading Style Sheets)**

CSS is a stylesheet language used to control the **design, layout, and appearance** of HTML pages.

✔ **Features:**

1. Controls colors, fonts, background, spacing
2. Helps create responsive web designs
3. Supports animations and transitions
4. Allows separation of content and design
5. Reusable code through external files

✔ **Importance:**

- Makes webpages attractive and consistent
- Saves time (change once → applied everywhere)
- Reduces HTML clutter
- Improves user experience and readability

22. Write the syntax of CSS declaration with example.

✔ **Syntax:**

```
selector {
  property: value;
}
```

✔ **Example:**

```
p {
  color: blue;
  font-size: 20px;
}
```

23. Explain different ways of using CSS in an HTML page.

✔ 1. Inline CSS

Written inside HTML tag.

```
<p style="color:red;">Hello</p>
```

✔ 2. Internal CSS

Inside <style> tag in <head>.

```
<style>
h1 { color: green; }
</style>
```

✔ 3. External CSS

Separate .css file linked to HTML.

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

24. What is an external CSS file? Write a sample CSS file.

An **external CSS file** is a separate .css file containing CSS rules.
It keeps HTML clean and allows reusability across multiple pages.

✔ Sample CSS file: style.css

```
body {
  background-color: lightgray;
}

h1 {
  color: blue;
}

p {
  font-size: 18px;
}
```

25. Define and explain selectors in CSS with examples.

Selectors are patterns used to select HTML elements for styling.

✔ **Types of Selectors:**

1. **Element Selector**

p { color: red; }

2. **Class Selector**

.myClass { font-size: 20px; }

3. **ID Selector**

#unique { background: yellow; }

4. **Universal Selector**

* { margin: 0; padding: 0; }

26. What is the CSS box model? Explain its components.

Every element in CSS is treated as a **box**, consisting of:

✔ **Components:**

1. **Content** → Actual text/image
2. **Padding** → Space inside the border
3. **Border** → Surrounds padding
4. **Margin** → Space outside the border

| Margin |
| Border |
| Padding |
| Content |

It helps control layout, spacing, and alignment.

27. What is padding in CSS? How can it be applied?

Padding is the **space between the content and the border** of an element.

✔ **Apply Padding:**

padding: 20px;

✔ **Different Padding Values:**

padding: 10px 15px 20px 25px;

(top, right, bottom, left)

28. Explain float and clear properties with examples.

✔ **float**

Positions an element left or right.

```
img {
  float: left;
}
```

✔ **clear**

Prevents elements from wrapping around floated items.

```
div {
  clear: both;
}
```

29. What is z-index in CSS? How does it work?

z-index controls **which element appears on top** when elements overlap.

✔ **Works only with positioned elements:**

(position: relative, absolute, fixed)

✔ **Example:**

```
div {
  position: absolute;
  z-index: 10;
}
```

Higher z-index = appears above others.

30. Write the benefits of using CSS in websites.

1. Better design and visual appeal
2. Faster page load time
3. Consistency across all pages
4. Easy maintenance
5. Reusable code
6. Creates responsive layouts

31. Explain how CSS helps in faster page loading and better design.

✔ **Faster Page Loading**

- External CSS loads once and is cached
- Reduces HTML file size
- Less inline styling → smaller code

✔ **Better Design**

- Advanced styling (colors, fonts, animations)
- Layout control via box model
- Responsive design using media queries

32. Differentiate between class selector and ID selector in CSS.

Class Selector	**ID Selector**
Used for multiple elements	Used for unique element
Starts with .	Starts with #
Less priority	Higher priority
Example: .item	Example: #header

33. How does padding affect the layout of an element?

Padding:

- Increases space **inside** the element
- Makes elements bigger
- Affects height/width calculations
- Pushes content away from borders

Example:

padding: 20px;

→ Add 20px inside border

34. Describe how to control the stacking order of elements using CSS.

Use **z-index** with positioning.

```
.box1 {
  position: absolute;
  z-index: 5;
}

.box2 {
  position: absolute;
  z-index: 10;
}
```

Element with **z-index 10** will appear on top.

35. Explain the importance of separating content and design using CSS.

Separating content (HTML) and design (CSS):

- Makes code clean and understandable
- Easy to update design without touching HTML
- Improves website performance
- Better SEO
- Allows consistent styling across all pages
- Simplifies maintenance for large websites

CHAPTER-IV CLIENT SIDE SCRIPTING

1. Explain Client-side scripting VS Server-side scripting with examples.

✔ **Client-side Scripting**

- Runs in the **browser**
- Code is visible to the user
- Faster execution
- Used for UI interactions, validation
- Example languages: **JavaScript**, HTML, CSS

Example:
JavaScript form validation in browser.

✔ **Server-side Scripting**

- Runs on **server**
- User cannot see code
- Used for database operations, login, backend logic
- Example languages: **PHP, Python, Node.js, Java**

Example:
Server checks login credentials from database.

2. Explain JavaScript basic syntax rules with suitable examples.

1. Statements end with **semicolon**
   let x = 10;
2. Case-sensitive
   Name ≠ name
3. Curly braces for blocks
4. if (x > 10) {
5.   console.log("OK");
6. }
7. Comments
   // single-line
   /* multi-line */

3. What are variables & identifiers? Explain naming rules with examples.

✔ **Variable**

Used to store data.

let age = 20;

✔ **Identifier**

Name given to a variable, function, etc.

✔ **Naming Rules**

- Cannot start with number → ✕ 1name
- Cannot use space → ✕ full name
- Only _ or $ allowed at start
- Case-sensitive
- Cannot use reserved words (like function, var)

Correct examples:

let myName;
let $price;
let _count;

4. Write a note on JavaScript data types & values with examples.

✔ **Primitive Data Types**

1. **Number**
   let x = 10;
2. **String**
   "Hello"
3. **Boolean**
   true, false
4. **Undefined**
   variable declared but no value
5. **Null**
   empty value
6. **Symbol**

7. **BigInt**

✔ **Non-Primitive Data Types**

- **Object**, **Array**, **Function**

Example:

```
let student = { name: "Atul", age: 20 };
let arr = [1,2,3];
```

5. Explain Scope in JavaScript (global vs function vs block scope) with example.

✔ **Global Scope**

Accessible everywhere.

```
var a = 10; // global
```

✔ **Function Scope**

Variables inside a function.

```
function test() {
  let x = 5; // function scope
}
```

✔ **Block Scope**

Variables inside { }
Only with **let** and **const**.

```
{
  let y = 20;
}
```

6. What are Literals in JavaScript? Explain its types with examples.

Literals = fixed values in code.

Types:

- **Numeric literal:** 100
- **String literal:** "Atul"
- **Boolean literal:** true

- **Array literal:** [1,2,3]
- **Object literal:** {name:"Aanya", age:17}
- **Null literal:** null

7. Write short note on Reserved Words in JavaScript.

Reserved words have special meaning in JS.
Cannot be used as variable names.

Examples:
var, let, const, if, else, switch, function, return, for, while, class, new

8. Explain Operators in JavaScript with examples.

✔ Arithmetic Operators

+ - * / % **
Example: 10 + 5

✔ Relational Operators

> < >= <= == !=
Example: x > y

✔ Logical Operators

&& || !
Example: (age > 18 && age < 60)

✔ Assignment Operators

= += -= *= /=
Example: x += 5;

9. Explain Control Statements in JavaScript (if, switch, loops) with examples.

✔ if Statement

```
if (age > 18) {
  console.log("Adult");
}
```

✔ switch Statement

```
switch(day) {
  case 1: console.log("Mon"); break;
}
```

✔ **Loops**

- **for loop**

```
for(let i=0;i<5;i++) console.log(i);
```

- **while loop**

```
while(x<5) x++;
```

- **do-while loop**

10. Explain Functions in JavaScript (User-defined, Built-in). Also explain parameters & return.

✔ **User-defined Function**

```
function greet(name) {
  return "Hello " + name;
}
```

✔ **Built-in Functions**

parseInt(), alert(), Math.random()

✔ **Parameters**

Values passed into a function.

✔ **Return**

Sends value back.

11. Explain JavaScript Objects. Explain Math / String / Date Object with methods.

✔ **JavaScript Object**

Collection of key-value pairs.

```
let person = {name:"Atul", age:20};
```

✔ **Math Object**

```
Math.sqrt(25)
Math.random()
Math.floor(4.8)
```

✔ **String Object**

```
"Atul".toUpperCase()
"Hello".length
"Hi there".includes("Hi")
```

✔ **Date Object**

```
let d = new Date();
d.getFullYear();
d.getMonth();
d.getDate();
```

12. Write short note on Regular Expressions with examples.

Regular expressions (RegEx) are patterns to match text.

✔ **Example:**

Check if email is valid:

```
let re = /^[a-z0-9]+@[a-z]+\.[a-z]{2,3}$/;
```

Match digits:

```
/[0-9]+/
```

13. Explain DOM in JavaScript. How to access & change HTML elements using DOM?

DOM (Document Object Model) represents HTML as a tree.

✔ **Access Elements:**

```
document.getElementById("title")
document.querySelector(".class")
```

✔ **Change Content:**

```
document.getElementById("title").innerHTML = "Welcome!";
```

✔ **Change Style:**

```
document.getElementById("title").style.color = "red"
```

14. Explain Event Handling in JavaScript with example.

Events = actions by user.

✔ **onclick()**

```
<button onclick="alert('Clicked!')">Click Me</button>
```

✔ **onmouseover()**

```
<p onmouseover="this.style.color='red'">Hover me</p>
```

✔ **onsubmit()**

```
<form onsubmit="return check()">...</form>
```

15. Write JavaScript code to validate an HTML form (email / mobile / password).

✔ **Example Validation:**

```
function validate() {
  let email = document.getElementById("email").value;
  let mobile = document.getElementById("mobile").value;

  let emailRe = /^[a-z0-9]+@[a-z]+\.[a-z]{2,3}$/;
  let mobRe = /^[0-9]{10}$/;

  if(!emailRe.test(email)) {
    alert("Invalid Email");
    return false;
  }

  if(!mobRe.test(mobile)) {
    alert("Invalid Mobile");
    return false;
  }
  return true;
}
```

16. Explain JavaScript variables, data types and values with examples.

✔ Variables

Variables are containers used to store data values in JavaScript.

Example:

```
let name = "Atul";
var age = 20;
const PI = 3.14;
```

✔ Data Types

JavaScript supports **two categories**:

◈ *Primitive Data Types*

1. **Number** → let x = 50;
2. **String** → "Hello"
3. **Boolean** → true / false
4. **Undefined** → declared but no value
5. **Null** → empty value
6. **Symbol**
7. **BigInt**

◈ *Non-Primitive (Reference) Types*

1. **Object** → let student = {name:"Aanya", age:17};
2. **Array** → [1,2,3]
3. **Function**

✔ Values

Values are actual data stored inside variables.

Example:

```
let marks = 90;       // Number
let msg = "Welcome";  // String
let active = true;    // Boolean
```

17. Explain scope in JavaScript (global, local, block) with examples.

✔ 1. Global Scope

Variables accessible **throughout the entire program**.

```
var x = 10;   // global

function show() {
  console.log(x);   // accessible
}
```

✔ 2. Local Scope (Function Scope)

Variables declared **inside a function**.

```
function test() {
  let a = 5;  // local
}
console.log(a); // ✕ Error
```

✔ 3. Block Scope

Available only inside { }
Uses **let** or **const**.

```
{
  let y = 20;
}
console.log(y); // ✕ Error
```

18. Explain JavaScript operators (arithmetic, relational, logical, assignment) with examples.

✔ 1. Arithmetic Operators

```
+ - * / % **
```

```
let a = 10 + 5;   // 15
```

✔ 2. Relational (Comparison) Operators

```
> < >= <= == === !=
10 > 5   // true
```

✔ 3. Logical Operators

```
&& || !
```

```
(a > 5 && a < 20)
```

✔ 4. Assignment Operators

```
= += -= *= /= %=
```

```
let x = 10;
x += 5;   // 15
```

19. Explain DOM & Event Handling in JavaScript with suitable example.

✔ DOM (Document Object Model)

DOM represents the HTML page as a **tree of elements**.
Using JavaScript, we can **access and modify** HTML.

◈ Accessing Elements

```
document.getElementById("title")
document.querySelector(".box")
```

◈ Changing HTML Content

```
document.getElementById("title").innerHTML = "Hello Atul!";
```

✔ Event Handling

Event handling means performing actions when a user interacts with an element.

◈ Example: onclick Event

```
<button onclick="msg()">Click Me</button>

<script>
function msg() {
  alert("Button Clicked!");
}
</script>
```

💠 onmouseover Event

```
<p onmouseover="this.style.color='red'">Hover over me</p>
```

💠 onsubmit Event

```
<form onsubmit="return check()">
```

20. Explain JavaScript Objects. Also explain Math, String and Date objects with methods.

✔ JavaScript Object

An object is a collection of **key–value pairs**.

```
let person = {
  name: "Atul",
  age: 20
};
```

☆ SHORT QUESTIONS (2 Marks / MCQ Type)

1. What is difference between let, var, const?

var	let	const
function scoped	block scoped	block scoped
can be redeclared	cannot be redeclared	cannot be redeclared
value change allowed	allowed	not allowed

2. What is NaN?

NaN = Not a Number
Represents an invalid number value.
Example: "abc" / 2 → NaN

3. What is typeof operator?

Used to check data type of value.

```
typeof 10  → "number"
```

typeof "hi" → "string"

4. What is callback function?

A function passed as **argument** to another function.

Example:

setTimeout(function(){ alert("Hello!"); }, 1000);

5. Use of strict equal (===).

=== checks **value + data type both**.

```
5 === "5"   // false
5 == "5"    // true
```

CHAPTER-V (JSON)

1. Explain JSON overview + why JSON is widely used in web applications? Compare JSON with XML.

✔ JSON (JavaScript Object Notation)

JSON is a lightweight data format used for storing and exchanging data between a client and a server.
It is text-based, human-readable, and easy for machines to parse.

✔ Why JSON is widely used?

- Very easy to read/write
- Faster than XML
- Uses less data (lightweight)
- Supported by all programming languages
- Works naturally with JavaScript
- Ideal for APIs and web services

☆ JSON vs XML

JSON	XML
Lightweight	Heavier
Uses key–value pairs	Uses tags

JSON	XML
Easy to read/write	More complex
Faster parsing	Slower parsing
No end tags	Uses opening + closing tags
Preferred in modern APIs	Older web systems

Example JSON:

{"name": "Atul", "age": 20}

Example XML:

<student><name>Atul</name><age>20</age></student>

2. Write JSON syntax rules. Explain proper key/value structure with example. Also write what is allowed & what is not allowed in JSON.

✔ JSON Syntax Rules

1. Data is in **key–value pairs**
2. Keys must be **strings in double quotes**
3. Values can be string, number, boolean, null, array, object
4. Data is separated by **commas**
5. Object is inside { }
6. Array is inside [ ]

✔ Proper Key–Value Example

```
{
 "name": "Atul",
 "age": 20,
 "isStudent": true
}
```

✔ Allowed in JSON

- Strings in **double quotes**
- Numbers
- true / false
- null

- Arrays
- Objects

❌ **NOT ALLOWED in JSON**

- Single quotes → 'Atul' ❌
- Trailing commas → "age":20, } ❌
- Comments (// or /**/) ❌
- Functions ❌

3. Explain JSON data types (String, Number, Boolean, Null, Object, Array) with examples.

✔ **1. String**

"name": "Atul"

✔ **2. Number**

"marks": 95

✔ **3. Boolean**

"isStudent": true

✔ **4. Null**

"middleName": null

✔ **5. Object**

"address": {"city": "Delhi", "pin": 110001}

✔ **6. Array**

"skills": ["HTML", "CSS", "JS"]

4. What is a JSON object? Explain objects vs arrays in JSON with examples.

✔ **JSON Object**

A collection of **key–value pairs** inside { }.

Example:

```
{
  "name": "Atul",
  "age": 20
}
```

✔ **Objects vs Arrays**

JSON Object	JSON Array
{ }	[ ]
Key-value pairs	Ordered list
Represents an entity	Represents a list
Example: student	Example: list of students

✔ **Examples:**

Object:

{"name": "Aanya", "age": 17}

Array:

["Aanya", "Atul", "Riya"]

Array of Objects:

```
[
  {"name":"Aanya", "age":17},
  {"name":"Atul", "age":20}
]
```

5. Explain JSON Schema. Why schema validation is needed? Write sample JSON schema for student record.

✔ **JSON Schema**

A JSON schema defines the **structure**, **data types**, and **rules** for a JSON document.

It ensures the data is:

- Correct
- Valid
- In proper format
- Safe

✔ **Why Validation Needed?**

- Prevents wrong data
- Ensures API receives expected format
- Helps debugging
- Avoids runtime errors

✔ **Sample JSON Schema for Student Record**

```
{
  "type": "object",
  "properties": {
    "name": {"type": "string"},
    "age": {"type": "number"},
    "roll": {"type": "number"},
    "isStudent": {"type": "boolean"}
  },
  "required": ["name", "age", "roll"]
}
```

6. How to convert (serialize) JavaScript object into JSON string? Explain JSON.stringify() with example.

✔ **JSON.stringify()**

Converts a JavaScript object → JSON string (serialization).

✔ **Example:**

```
let student = {
  name: "Atul",
  age: 20
};

let jsonText = JSON.stringify(student);
console.log(jsonText);
```

Output:

```
{"name":"Atul","age":20}
```

7. How to convert JSON string into JavaScript object? Explain JSON.parse() with example + error conditions.

✔ **JSON.parse()**

Converts JSON string → JavaScript object.

✔ **Example:**

```
let txt = '{"name":"Aanya","age":17}';
let obj = JSON.parse(txt);

console.log(obj.name);
```

✔ **Error Conditions**

- If JSON has **single quotes**, parse fails
- If JSON has trailing commas
- If JSON contains comments
- If JSON syntax is invalid

8. Explain JSON Object in detail with example.

A JSON Object is a set of **key-value pairs** enclosed in **{ }**.

✔ **Example:**

```
{
 "id": 101,
 "name": "Atul",
 "skills": ["HTML", "CSS", "JS"],
 "address": {
   "city": "Delhi",
   "pincode": 110001
 }
}
```

This object contains:

- number
- string
- array

- another object

9. Explain data type of JSON in detail.

JSON supports **6 basic data types**:

✔ 1. String

Text inside double quotes
"hello"

✔ 2. Number

No separate int/float
10, 20.5

✔ 3. Boolean

true or false

✔ 4. Null

Represents empty
null

✔ 5. Object

Key-value pairs
{"name":"Atul"}

✔ 6. Array

Ordered list
["pen","book"]

FWT EXAM PREPRATION

CH.1 INTRODUCTION TO WEB TECHNOLOGY

1) Define Internet and WWW.

Internet:
The Internet is a global network of interconnected computers that communicate using standard protocols. It allows users to share information, access resources, and connect worldwide.

WWW (World Wide Web):
The WWW is a collection of web pages and web resources accessed through the Internet using a web browser. It uses hyperlinks and URLs to navigate between pages.

2) What is HTTP? Write its full form.

HTTP (HyperText Transfer Protocol) is a communication protocol used for transferring web pages from a web server to a web browser.

3) What is the difference between HTTP and HTTPS?

- **HTTP:** Unsecured communication; data is sent as plain text.
- **HTTPS:** Secured communication using encryption (SSL/TLS). Data is safe from attackers.

4) Define a web browser.

A **web browser** is a software application used to access, view, and browse web pages on the World Wide Web.

5) What is a URL? Give an example.

A **URL (Uniform Resource Locator)** is the address used to locate a resource on the Internet.
Example: https://www.google.com

6) List any two functions of a web client.

1. Sends HTTP requests to web servers.
2. Displays web pages received from servers.

7) Define FTP and state its use.

FTP (File Transfer Protocol) is a protocol used to upload and download files between a client and a server on the Internet.

8) Compare HTTP and HTTPS with an example.

HTTP	HTTPS
Not encrypted	Encrypted
URL starts with **http://**	URL starts with **https://**
Less secure	Highly secure

Example:
http://example.com vs https://example.com

9) Explain the structure of a URL.

A URL has these main parts:

1. **Protocol** – e.g., https
2. **Domain name** – e.g., google.com
3. **Path** – e.g., /search
4. **Parameters** – e.g., ?q=chatgpt

Example: https://www.example.com/page?id=10

10) What are the major functions of a web browser?

- Rendering and displaying web pages
- Managing bookmarks
- Handling cookies and cache
- Executing JavaScript
- Ensuring secure browsing (HTTPS support)

11) Differentiate between Internet and WWW.

- **Internet:** Physical network of computers.
- **WWW:** Collection of web pages stored on the Internet.

12) Explain the working of FTP.

FTP works by establishing a connection between an FTP client and an FTP server.

- The client sends a request to download/upload files.
- The server authenticates the user.
- Data transfer occurs through data and control channels.

13) Explain Internet, WWW, HTTP, and HTTPS in detail with examples.

Internet: A global network connecting millions of devices.
WWW: A system of interlinked hypertext documents accessed via the Internet.
HTTP: Protocol for transferring web pages.
HTTPS: Secure version of HTTP with encryption.

Example:
https://en.wikipedia.org loads via HTTPS from the WWW over the Internet.

14) What is a web browser? Explain its components and functionalities.

A web browser is software to access the web.

Components:

- **User Interface**
- **Rendering Engine**
- **JavaScript Engine**
- **Networking Module**
- **Data Storage (cache, cookies)**

Functionalities:

- Displays web pages
- Handles security
- Loads HTML, CSS, JS
- Downloads files

15) What is FTP? Describe how file transfer occurs using FTP.

FTP allows file transfer between client and server.
Transfer steps:

1. Client connects to server
2. Login/authentication

3. File upload/download
4. Server sends confirmation

16) Describe the architecture of a web client and its role in web communication.

A web client consists of:

- **User Interface**
- **Networking Module**
- **Rendering Engine**
- **Scripting Engine**

Role:

- Sends HTTP requests
- Receives and displays responses
- Handles cookies, cache
- Makes communication possible with servers

17) Discuss the difference between URL and URI with proper format.

- **URL (Locator):** Location of a resource.
  Example: https://example.com/about
- **URI (Identifier):** Can identify a resource even without giving a location.
  Example: mailto:info@example.com

18) What is a web server?

A **web server** is software or hardware that stores web pages and delivers them to clients on request.

19) List any two features of a web server.

1. Handles HTTP requests
2. Provides security and access control

20) What is access control in a web server?

Access control restricts which users or systems can access specific files or resources on the server.

21) What is server-side logging?

Server-side logging records activities such as user requests, errors, and server status for monitoring and troubleshooting.

22) Write a short note on the history of web servers.

The first web server was created by Tim Berners-Lee in 1990 (CERN).
Later, Apache became the most popular server in the early 2000s.
Modern servers include Nginx, IIS, and cloud-based services.

23) Explain any three features of a web server.

1. **Request Handling**
2. **Security Support (HTTPS)**
3. **Logging and Monitoring**

24) What is the use of logging in web servers?

Logging helps administrators track errors, performance, and user activity. It is essential for security audits and troubleshooting.

25) Define access control. How does it secure a web server?

Access control ensures only authorized users can access resources.
It improves security by preventing unauthorized access or attacks.

26) Explain the main features of a web server in detail.

- Request processing
- Content management
- Security support
- Logging
- Session handling
- Scalability features

27) Describe the history and evolution of web servers.

- 1990: First server at CERN
- 1995: Apache introduced
- 2004: Nginx for high performance
- Modern era: Cloud servers, load balancers, container-based servers

28) What is logging in a web server? How is it useful in administration?

Logging records all server activities.
Helps in monitoring performance, detecting attacks, analyzing traffic, and fixing issues.

29) Discuss access control mechanisms used in web servers.

1. **Password-based access**
2. **IP-based restrictions**
3. **Role-based access control (RBAC)**
4. **SSL certificates**

30) Explain the working of a web server with a labeled diagram.

Working:

1. Client sends HTTP request
2. Server processes request
3. Server fetches required file/page
4. Server sends back response
5. Browser displays page

31. Differentiate between HTTP and HTTPS with examples.

HTTP (HyperText Transfer Protocol):

- Used for transferring web pages over the Internet.
- The data is *not encrypted* and can be intercepted.
- Suitable for general browsing where security is not critical.
- URL starts with **http://**

HTTPS (HyperText Transfer Protocol Secure):

- Secure version of HTTP that uses **SSL/TLS encryption**.
- Protects data from hackers during transmission.
- Used for banking, payments, login pages, etc.
- URL starts with **https://**

Example:

- HTTP → http://example.com
- HTTPS → https://example.com (secure)

32. Explain URL and its components with example.

A **URL (Uniform Resource Locator)** is the address used to access a resource on the Internet.

Components of a URL:

1. **Protocol** – Defines communication type (e.g., https, ftp)
2. **Domain Name** – Name of the website (e.g., google.com)
3. **Port Number** (optional) – e.g., :80, :443
4. **Path** – Specific location/resource on the server (e.g., /products)
5. **Query Parameters** (optional) – Extra information (e.g., ?id=10)

Example:

https://www.amazon.com/electronics?category=mobile

- Protocol → https
- Domain → amazon.com
- Path → /electronics
- Query → ?category=mobile

33. What is the difference between Internet and World Wide Web (WWW)?

Internet:

- A global network of interconnected computers.
- Provides communication services like email, file sharing, video calls, etc.
- Hardware + network infrastructure.

WWW:

- A collection of web pages, websites, and multimedia content accessible using the Internet.
- Uses HTTP/HTTPS protocols.
- Part of the Internet, not the whole Internet.

In short:
Internet = physical network
WWW = information on that network

34. What are the functions of a web browser?

1. Fetches web pages from web servers using HTTP/HTTPS
2. Renders HTML, CSS, and JavaScript to display webpages
3. Provides user interface (tabs, back/forward, bookmarks, history)
4. Manages cookies, cache, and session data
5. Ensures security using SSL certificates
6. Allows downloading files and saving webpages

35. Explain FTP and how it works for file transfer.

FTP (File Transfer Protocol):
A protocol used to upload/download files between a client and a server over a network.

How FTP works:

1. **Connection Establishment:**
   FTP client connects to the FTP server using the server address.
2. **Authentication:**
   User logs in using a username and password (or anonymous login).
3. **File Transfer Mode:**
   - ASCII mode → for text files
   - Binary mode → for images, videos, software
4. **Transfer Process:**
   The client sends commands and the server sends back files using separate control and data channels.
5. **Completion:**
   The server confirms successful upload/download.

36. What is a web server? Explain its working.

A **web server** is software or hardware that stores web pages and delivers them to clients on request.

Working of a Web Server:

1. **Client Request:**
   Browser sends an HTTP/HTTPS request to the server.
2. **Processing:**
   The server checks the request, locates the requested file/page, or executes scripts if needed.

3. **Response:**
   The server sends back an HTML page or data along with a status code (e.g., 200 OK).
4. **Rendering:**
   Browser displays the content to the user.

37. Write a short note on the features of a web server.

1. **Request Handling:**
   Processes client HTTP/HTTPS requests efficiently.
2. **Security Support:**
   Supports SSL/TLS for secure communication.
3. **Logging:**
   Records user activity, errors, and server performance.
4. **Load Handling:**
   Can manage multiple users at the same time.
5. **Content Management:**
   Stores and serves web pages, images, scripts, and files.

38. Explain the concept of server-side logging. Why is it important?

Server-side logging refers to recording all significant activities performed on the server, such as requests, responses, errors, and user actions.

Importance of server-side logging:

- Helps detect errors and troubleshoot issues
- Monitors server performance
- Helps identify security threats and unauthorized access
- Useful for auditing and analyzing website traffic
- Ensures smooth administration of the server

39. What is access control in a web server? How does it enhance security?

Access Control is a security mechanism used to restrict who can access certain files, directories, or resources on a web server.

How it enhances security:

- Prevents unauthorized users from accessing sensitive data
- Protects admin panels and confidential files
- Ensures only authenticated users can access specific areas
- Helps stop hacking attempts and misuse

Types:

- Password-based access
- IP-based restrictions
- Role-Based Access Control (RBAC)

40. Explain the history and evolution of web servers.

- **1990:**
  The first web server (CERN httpd) was created by Tim Berners-Lee at CERN.
- **Mid-1990s:**
  Apache HTTP Server released → became the most popular server due to open-source support.
- **2000s:**
  Microsoft IIS (Internet Information Services) gained popularity on Windows platforms.
- **2004:**
  Nginx introduced → optimized for speed and handling high-traffic websites.
- **Present:**
  Modern servers include cloud-based systems like AWS, Google Cloud, and containerized servers using Docker & Kubernetes.

Evolution trend:
Basic text servers → dynamic content → secure HTTPS → scalable cloud servers

CH.2 HTML, XHTML & HTML5

1. Explain the structure of an HTML document with a neat diagram.

An HTML document follows a proper structure that helps the browser understand the content.

Structure:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>

  <body>
    <h1>Main Content</h1>
    <p>This is a paragraph.</p>
  </body>
</html>
```

Explanation of Sections:

- **<!DOCTYPE html>** → Declares HTML5 document
- **<html>** → Root element
- **<head>** → Contains metadata (title, meta tags, styles)
- **<title>** → Name of the webpage
- **<body>** → Displays all visible content

Neat Diagram (ASCII):

```
HTML Document
 ├── <!DOCTYPE html>
 └── <html>
      ├── <head>
      │    ├── <title>
      │    └── Metadata
      └── <body>
           ├── Headings
           ├── Paragraphs
           ├── Images
           └── Links
```

2. What are lists in HTML? Explain ordered, unordered, and definition lists with examples.

HTML lists are used to display items in a structured way.

1. Ordered List (<ol>) – numbered list

```
<ol>
  <li>Apple</li>
  <li>Banana</li>
  <li>Mango</li>
</ol>
```

2. Unordered List (<ul>) – bullet list

```
<ul>
  <li>Pen</li>
  <li>Pencil</li>
  <li>Book</li>
</ul>
```

3. Definition List (<dl>) – term & definition

```
<dl>
  <dt>HTML</dt>
  <dd>HyperText Markup Language</dd>
</dl>
```

3. Explain the navigation structure of a website. Write HTML code to open a link in a new tab/window.

Navigation structure represents how users move from one page to another.
It usually contains menus like:

- Home
- About
- Services
- Contact

A navigation menu is created using **links (<a>)**.

HTML code to open a link in a new tab:

```
<a href="https://www.google.com" target="_blank">Open Google</a>
```

target="_blank" opens link in a new tab/window.

4. What is an HTML form? Explain <form> **attributes and design a student registration form.**

HTML Form:

A form collects user input and sends it to a server.

Important <form> Attributes:

- **action** → URL to send form data
- **method** → GET or POST
- **enctype** → Type of data encoding
- **name** → Name of the form
- **target** → Where to display result

Student Registration Form:

```
<form action="submit.php" method="post">
  <h3>Student Registration</h3>

  Name: <input type="text" name="name"><br><br>
  Email: <input type="email" name="email"><br><br>
  Age: <input type="number" name="age"><br><br>

  Gender:
  <input type="radio" name="gender" value="Male"> Male
  <input type="radio" name="gender" value="Female"> Female<br><br>

  Course:
  <select name="course">
    <option>BCA</option>
    <option>BBA</option>
  </select><br><br>

  <input type="submit" value="Register">
</form>
```

5. Write HTML code to create a login page with username, password, and submit button.

```
<form>
  <h2>Login Page</h2>

  Username:
  <input type="text" name="username"><br><br>

  Password:
  <input type="password" name="password"><br><br>

  <input type="submit" value="Login">
</form>
```

6. Explain HTML tables in detail. Write code for a timetable using rowspan and colspan.

HTML Table Concepts:

- **<table>** → table container
- **<tr>** → table row
- **<td>** → table data
- **<th>** → table heading
- **rowspan** → merge cells vertically
- **colspan** → merge cells horizontally

Time-Table Example:

```
<table border="1">
  <tr>
    <th colspan="5">Time Table</th>
  </tr>

  <tr>
    <th>Day</th>
    <th>9-10</th>
    <th>10-11</th>
    <th>11-12</th>
    <th>12-1</th>
  </tr>

  <tr>
```

```
    <td rowspan="2">Monday</td>
    <td>Math</td>
    <td>English</td>
    <td rowspan="2">Break</td>
    <td>Science</td>
  </tr>

  <tr>
    <td>Computer</td>
    <td>Sports</td>
    <td>Arts</td>
  </tr>
</table>
```

7. Differentiate between HTML, XHTML, and HTML5 with examples.

HTML

- Flexible, not strict
- Example: Tags may be unclosed
- <p> Hello

XHTML

- Stricter, XML-based
- All tags must be closed
- <br />, <img src="a.jpg" />

HTML5

- Modern version with new features
- Audio/video support, semantic tags
- <video controls> … </video>

8. Explain the new features of HTML5 with examples.

1. **Audio & Video support**
   <audio controls src="song.mp3"></audio>
2. **Canvas for graphics**
3. **Semantic tags** → <header>, <nav>, <footer>
4. **New form input types** → email, date, range
5. **Local Storage & Session Storage**
6. **Geolocation API**

9. What are semantic elements in HTML5? Explain with suitable examples.

Semantic elements describe meaning of content.

Examples:

- **<header>** → Top section
- **<nav>** → Navigation menu
- **<article>** → Independent article
- **<section>** → Logical section
- **<footer>** → Bottom section

Example Code:

```
<header>My Website</header>
<nav>Home | About | Contact</nav>
<section>
  <article>Article content here</article>
</section>
```

10. Write HTML code for a feedback form with text, radio buttons, checkboxes, and reset button.

```
<form>
  <h3>Feedback Form</h3>

  Name:
  <input type="text" name="name"><br><br>

  Rate Us:
  <input type="radio" name="rate" value="Good"> Good
  <input type="radio" name="rate" value="Average"> Average
  <input type="radio" name="rate" value="Bad"> Bad<br><br>

  Services Used:<br>
  <input type="checkbox" name="service" value="Support"> Support<br>
  <input type="checkbox" name="service" value="Delivery"> Delivery<br>
  <input type="checkbox" name="service" value="Website"> Website<br><br>

  <input type="reset" value="Reset Form">
  <input type="submit" value="Submit">
</form>
```

11. Explain Block-level and Inline elements in XHTML with examples.

Block-level Elements

- Start on a **new line**
- Occupy the **full width** available
- Can contain inline elements
- Examples:
  <div>, <p>, <h1>–<h6>, <ul>, <table>

Example:

```
<p>This is a block element.</p>
```

Inline Elements

- Do **not** start on a new line
- Occupy only the **required width**
- Usually used inside block elements
- Examples:
  <span>, <a>, <img>, <strong>, <small>

Example:

```
<p>This is <span>inline text</span> inside a paragraph.</p>
```

12. Write HTML5 code to embed an audio and a video file with controls.

```
<h3>Audio Example</h3>
<audio controls>
  <source src="song.mp3" type="audio/mpeg">
</audio>

<h3>Video Example</h3>
<video controls width="400">
  <source src="movie.mp4" type="video/mp4">
</video>
```

13. Compare HTML4 and HTML5 with features and examples.

HTML4

- Older version
- No built-in multimedia support

- Depends on external plugins like Flash
- No semantic elements

Example:
<div id="header">

HTML5

- New modern version
- Supports **audio**, **video**, **canvas**
- Semantic tags → <header> <section> <footer>
- Better form controls

Example:
<header>Website Title</header>

Major Differences:

HTML4	**HTML5**
No audio/video tags	Built-in multimedia support
No semantic tags	Semantic tags included
Strict Doctype	Simple DOCTYPE
Needs Flash	Works without Flash

14. What are deprecated tags in HTML5? List and explain with alternatives.

Deprecated tags are **removed or outdated tags** that should not be used in HTML5.

Deprecated Tags & Alternatives

Deprecated Tag	Meaning	HTML5 Alternative
<center>	Center text	text-align:center; (CSS)
<font>	Change font style	CSS font-family, color
<big> / <small>	Font size	CSS font-size

Deprecated Tag	Meaning	HTML5 Alternative
<frame>, <frameset>	Frames layout	<iframe> or CSS layout
<u>	Underline	CSS text-decoration: underline;

15. Explain the use of <meta> **tags (charset, description, keywords, refresh). Give examples.**

<meta> tags provide metadata to the browser.

1. charset

Defines character encoding.

<meta charset="UTF-8">

2. description

Describes webpage content for search engines.

<meta name="description" content="This is my website.">

3. keywords

Keywords for SEO.

<meta name="keywords" content="HTML, CSS, Web Development">

4. refresh

Reloads page automatically.

<meta http-equiv="refresh" content="5">

16. Explain frames in HTML with example. Why are they removed in HTML5?

Frames in HTML

Frames divide the browser window into multiple sections.

Example:

```
<frameset cols="50%,50%">
  <frame src="left.html">
  <frame src="right.html">
</frameset>
```

Why Frames Removed in HTML5?

- Not mobile-friendly
- Not good for SEO
- Difficult navigation
- Complex usability
- Security issues

HTML5 uses **CSS layouts**, **iframe**, and **Flexbox/Grid** instead.

17. Write HTML code for a navigation menu using anchor and list tags.

```
<ul>
  <li><a href="home.html">Home</a></li>
  <li><a href="about.html">About</a></li>
  <li><a href="services.html">Services</a></li>
  <li><a href="contact.html">Contact</a></li>
</ul>
```

18. Explain difference between <div> **and** <span> **with examples.**

<div>

- Block-level element
- Used for large sections
- Starts on a new line

Example:

```
<div>
  <h2>Content Section</h2>
</div>
```

<span>

- Inline element
- Used for small text formatting
- Does not start on a new line

Example:

```
<p>This is <span style="color:red;">highlighted</span> text.</p>
```

19. Discuss the role of HTML Validator. Why is it important in XHTML?

HTML Validator

A tool that checks HTML code for:

- Errors
- Invalid tags
- Missing closing tags
- Syntax issues

Importance in XHTML

- XHTML is strict (XML-based)
- Requires proper nesting and closing of tags
- Validator ensures:
  - Cleaner code
  - Browser compatibility
  - Better SEO
  - Fewer runtime errors

20. Write HTML code for a resume web page (with headings, lists, table, links).

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

<body>
  <h1>Atul Pal</h1>
  <h2>Contact</h2>
  <p>Email: atul@example.com</p>
  <a href="https://linkedin.com">LinkedIn Profile</a>

  <h2>Skills</h2>
  <ul>
    <li>HTML</li>
    <li>CSS</li>
    <li>C Programming</li>
  </ul>
  <h2>Education</h2>
  <table border="1">
    <tr>
      <th>Year</th>
      <th>Course</th>
      <th>Institute</th>
    </tr>
    <tr>
      <td>2024</td>
      <td>BCA</td>
      <td>GTU</td>
    </tr>
  </table>
</body>
</html>
```

21. Define HTML. List its advantages and limitations.

HTML (HyperText Markup Language)

HTML is a markup language used to create and display web pages. It uses tags to define headings, paragraphs, images, tables, forms, etc.

Advantages of HTML

1. Easy to learn and use
2. Supported by all browsers
3. Free and platform-independent
4. Integrates with CSS & JavaScript
5. Lightweight and fast to load

Limitations of HTML

1. Not secure (only a structure, no logic)
2. Cannot create dynamic pages without CSS/JS
3. Limited styling options (depends on CSS)
4. Cannot interact with databases directly
5. Difficult to maintain large HTML files

22. Explain the difference between absolute and relative hyperlinks with examples.

1. Absolute Hyperlink

- Contains complete URL
- Points to a webpage anywhere on the Internet
  Example:

```
<a href="https://www.google.com">Google</a>
```

2. Relative Hyperlink

- Points to a page within the same website
- Uses only file path, not full URL
  Example:

```
<a href="about.html">About Us</a>
```

23. Write HTML code to display 3 images in a single row.

```
<img src="img1.jpg" width="200">
<img src="img2.jpg" width="200">
<img src="img3.jpg" width="200">
```

(All <img> tags are inline elements, so they appear in one row.)

24. Write HTML code to create a list of subjects using <ol> **and** <ul>**.**

```
<h3>Ordered List</h3>
<ol>
  <li>Mathematics</li>
  <li>Physics</li>
  <li>Chemistry</li>
</ol>

<h3>Unordered List</h3>
<ul>
  <li>English</li>
  <li>Computer Science</li>
  <li>Biology</li>
</ul>
```

25. Explain the use of <head> **tag and its elements (**<title>**,** <meta>**,** <link>**).**

The **<head>** section contains metadata and information about the webpage.

1. <title> tag

- Displays the title on the browser tab

```
<title>My Webpage</title>
```

2. <meta> tag

- Provides metadata like keywords, description, charset

```
<meta charset="UTF-8">
<meta name="description" content="HTML Tutorial">
```

3. <link> tag

- Links external files (CSS, icons)

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

26. Write HTML code to display 5 headings and 3 paragraphs.

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>

<p>This is paragraph one.</p>
<p>This is paragraph two.</p>
<p>This is paragraph three.</p>
```

27. Explain the use of <br> **and** <hr> **with examples.**

1. <br> – Line Break

- Moves text to next line
- No closing tag
  Example:

```
Hello<br>World
```

2. <hr> – Horizontal Rule

- Creates a horizontal line
- Used to separate sections
  Example:

```
<h2>Title</h2>
<hr>
<p>Paragraph below line.</p>
```

28. Difference between HTML elements and attributes with examples.

HTML Elements

- Consist of start tag, content, and end tag
  Example:

```
<p>This is a paragraph.</p>
```

HTML Attributes

- Provide extra information inside the tag
  Example:

```
<img src="car.jpg" width="200">
```

Difference Table

HTML Elements	HTML Attributes
Define content	Add extra info
Have opening & closing tags	Exist inside tags
Example: <h1>	Example: src="image.jpg"

29. Write HTML code to create a form with text, password, and submit button.

```
<form>
  Username: <input type="text" name="user"><br><br>
  Password: <input type="password" name="pass"><br><br>
  <input type="submit" value="Login">
</form>
```

30. What is the difference between inline CSS and internal CSS in HTML?

Inline CSS

- Written inside the HTML tag
- Affects only one element
  Example:

```
<p style="color:red;">Hello</p>
```

Internal CSS

- Written inside <style> tag in <head>
- Affects entire page
  Example:

```
<head>
<style>
p { color: red; }
</style>
</head>
```

Difference Table

Inline CSS	Internal CSS
Applied to a single element	Applied to multiple elements
High priority	Medium priority
Hard to maintain	Easier to maintain

31. Write HTML code to create a hyperlink to another page.

```
<a href="about.html">Go to About Page</a>
```

32. Explain the difference between HTML and XHTML (rules of XHTML).

HTML

- Flexible
- Not strict
- Tags can be unclosed
- Case-insensitive

XHTML

- Stricter and XML-based
- All tags **must** be closed
- Lowercase tags only
- Attributes must have quotes
- Proper nesting required

Example Difference

HTML:

```
<br>
<img src="pic.jpg">
```

XHTML:

```
<br />
<img src="pic.jpg" alt="image" />
```

Rules of XHTML

1. Tags must be in lowercase
2. All tags must be closed
3. Attributes must be in quotes
4. Proper nesting required
5. Must include DOCTYPE for XHTML

33. What are input types in HTML5 forms? List any five with examples.

HTML5 introduced new input types:

Types with Examples:

1. **email**

```
<input type="email" name="mail">
```

2. **date**

```
<input type="date" name="dob">
```

3. **range**

```
<input type="range" min="0" max="10">
```

4. **number**

```
<input type="number" min="1" max="10">
```

5. **color**

```
<input type="color">
```

Other types: url, tel, search, time, datetime-local, week, month.

34. Write HTML code to create a table with a caption.

```
<table border="1">
  <caption>Student Marks</caption>

  <tr>
    <th>Name</th>
    <th>Marks</th>
  </tr>

  <tr>
    <td>Atul</td>
    <td>95</td>
  </tr>

  <tr>
    <td>Riya</td>
    <td>89</td>
  </tr>
</table>
```

35. Explain the use of <iframe> **with an example.**

<iframe>

- Embeds another webpage inside the current page
- Used for maps, ads, videos, external content

Example:

```
<iframe src="https://www.wikipedia.org" width="500" height="300"></iframe>
```

36. What is the difference between semantic and non-semantic tags in HTML5?

Semantic Tags

- Have meaningful names
- Clearly describe content
- Improve SEO and accessibility

Examples:
<header>, <footer>, <section>, <article>, <nav>

Non-Semantic Tags

- Do not describe meaning
- Only used for layout
- Less clear for machines

Examples:
<div>, <span>

37. List any five block-level elements and any five inline elements.

Block-level Elements

1. <div>
2. <p>
3. <h1>
4. <ul>
5. <table>

Inline Elements

1. <span>
2. <a>
3. <img>
4. <strong>
5. <em>

38. Write HTML code to display a YouTube video using <iframe>**.**

```
<iframe
  width="560"
  height="315"
  src="https://www.youtube.com/embed/VIDEO_ID"
  frameborder="0"
  allowfullscreen>
</iframe>
```

(Replace VIDEO_ID with actual YouTube video ID.)

39. Explain the purpose of DOCTYPE declaration in HTML/XHTML.

Purpose of DOCTYPE:

- Tells the browser which version of HTML/XHTML the document uses
- Ensures **standards mode**
- Prevents rendering issues
- Helps browser interpret tags correctly

HTML5 DOCTYPE:

```
<!DOCTYPE html>
```

XHTML DOCTYPE:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

40. List differences between HTML and DHTML.

HTML

- Static
- Cannot change content dynamically
- Only markup language
- No animations or interactions

DHTML (Dynamic HTML)

- Combines HTML + CSS + JavaScript
- Supports animation & dynamic effects
- Content can change without reloading
- Used for menus, slideshows, effects

Table:

HTML	DHTML
Static pages	Dynamic pages
No scripting	Uses JavaScript
Simple layout	Interactive layout
No animations	Supports animations

41. Explain the basic structure of an HTML document with an example.

An HTML document follows a fixed structure so the browser can properly display the page.

✔ **Structure:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page</title>
  </head>

  <body>
    <h1>Welcome</h1>
    <p>This is my webpage.</p>
  </body>
</html>
```

✔ **Explanation:**

- **<!DOCTYPE html>** → Declares HTML5 document
- **<html>** → Root element
- **<head>** → Contains metadata
- **<title>** → Title shown in browser tab
- **<body>** → All visible content appears here

42. What are lists in HTML? Explain ordered, unordered, and definition lists with examples.

HTML lists display items in a structured format.

✔ 1. Ordered List (Numbered)

```
<ol>
  <li>Apple</li>
  <li>Banana</li>
</ol>
```

✔ 2. Unordered List (Bulleted)

```
<ul>
  <li>Pen</li>
  <li>Pencil</li>
</ul>
```

✔ 3. Definition List

```
<dl>
  <dt>HTML</dt>
  <dd>HyperText Markup Language</dd>
</dl>
```

43. Differentiate between HTML and XHTML with suitable examples.

HTML	**XHTML**
Not strict	Very strict
Tags may remain unclosed	All tags must be closed
Attributes may not require quotes	All attributes must use quotes
Case-insensitive	Lowercase tags required

✔ Examples:

HTML:

```
<br>
```

<img src="pic.jpg">

XHTML:

<br />
<img src="pic.jpg" alt="image" />

44. Explain Block-level and Inline elements with examples.

✔ Block-level Elements

- Start on a new line
- Take full width
- Example: <div>, <p>, <h1>

<p>This is a block element.</p>

✔ Inline Elements

- Do not start on a new line
- Take only required width
- Example: <span>, <a>, <img>

<span>This is inline.</span>

45. What is a form in HTML? Explain different form elements with an example.

✔ HTML Form:

Used to collect user input.

✔ Common Form Elements:

- **<input type="text">** → Text box
- **<input type="radio">** → Radio button
- **<input type="checkbox">** → Checkbox
- **<textarea>** → Multi-line input
- **<select>** → Dropdown
- **<input type="submit">** → Submit button

✔ Example:

<form>

```
  Name: <input type="text"><br>
  Gender: <input type="radio">Male <input type="radio">Female<br>
  Hobbies: <input type="checkbox">Music<br>
  <input type="submit">
</form>
```

46. Explain HTML tables with an example showing <thead>, <tbody>, <tfoot>.

```
<table border="1">
  <thead>
    <tr><th>Name</th><th>Marks</th></tr>
  </thead>

  <tbody>
    <tr><td>Atul</td><td>90</td></tr>
    <tr><td>Riya</td><td>85</td></tr>
  </tbody>

  <tfoot>
    <tr><td>Total Students</td><td>2</td></tr>
  </tfoot>
</table>
```

47. Discuss the new features of HTML5 in detail.

1. **Audio & Video Support**
2. **Canvas** for graphics
3. **Semantic tags** → <header> <article> <footer>
4. **New input types** → email, date, range
5. **LocalStorage & SessionStorage**
6. **Geolocation API**
7. **Responsive design support**

48. Explain semantic elements in HTML5 with examples.

✔ **Examples:**

- **<header>** – Top section
- **<article>** – Independent article
- **<section>** – Group of related content
- **<footer>** – Bottom section

✔ **Example Code:**

```
<header>My Website</header>
<section>
  <article>News Article</article>
</section>
<footer>Copyright 2025</footer>
```

49. Write an HTML code to create a student registration form.

```
<form>
  Name: <input type="text"><br><br>

  Gender:
  <input type="radio" name="g">Male
  <input type="radio" name="g">Female<br><br>

  Hobbies:
  <input type="checkbox">Music
  <input type="checkbox">Sports<br><br>

  <input type="submit" value="Register">
</form>
```

50. Compare HTML4, XHTML, and HTML5.

HTML4	**XHTML**	**HTML5**
Old version	Stricter XML version	Modern version
No multimedia	Strict rules	Supports audio/video
No semantic tags	Must close tags	Semantic tags
Complex DOCTYPE	Long DOCTYPE	Simple DOCTYPE

51. Explain the use of <meta> and <title> tags.

✔ **<title>**

- Displays page title in browser

```
<title>My Page</title>
```

✔ **<meta>**

Provides metadata:

```
<meta charset="UTF-8">
<meta name="keywords" content="HTML, Web">
<meta name="description" content="Tutorial">
```

52. Differentiate between absolute and relative links with examples.

✔ **Absolute Link**

Full URL

```
<a href="https://google.com">Google</a>
```

✔ **Relative Link**

Link within same website

```
<a href="about.html">About</a>
```

53. List any five tags used in HTML with their purpose.

Tag	Purpose
<p>	Paragraph
<h1>	Heading
<img>	Insert image
<a>	Hyperlink
<table>	Create table

54. Write HTML code to create a table with 3 rows and 3 columns.

```
<table border="1">
  <tr><td>A</td><td>B</td><td>C</td></tr>
  <tr><td>D</td><td>E</td><td<F</td></tr>
  <tr><td>G</td><td>H</td><td>I</td></tr>
</table>
```

55. Explain the use of frames in HTML. Why are they deprecated in HTML5?

✔ **Use of Frames:**

Frames divide the window into multiple sections:

```
<frameset>
  <frame src="left.html">
  <frame src="right.html">
</frameset>
```

✔ **Deprecated Because:**

- Not mobile friendly
- Poor SEO
- Complicated navigation
- Security risks
- HTML5 uses CSS layout instead

56. What are HTML attributes? Explain with two examples.

Attributes give extra information to tags.

✔ **Examples:**

```
<img src="image.jpg" width="200">
<a href="home.html">Home</a>
```

57. Write HTML5 code to insert audio and video.

```
<audio controls>
  <source src="music.mp3" type="audio/mpeg">
</audio>

<video controls width="300">
  <source src="video.mp4" type="video/mp4">
</video>
```

58. Explain the difference between <div> and <span>.

<div>	<span>
Block element	Inline element
Takes full width	Takes only required width
Used for large sections	Used for small text styling

59. Write HTML code to display three headings and two paragraphs.

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>

<p>This is paragraph one.</p>
<p>This is paragraph two.</p>
```

60. List any five deprecated tags in HTML5.

1. <center>
2. <font>
3. <big>
4. <u>
5. <frameset>

CHAPTER 3

1. What do you mean by CSS declaration?

A **CSS declaration** defines how an HTML element should be styled.
It has **property : value** format.

Example:

```
color: red;
font-size: 20px;
```

2. Write the types of CSS with examples.

✔ **1. Inline CSS**

Written inside the HTML tag.

```
<p style="color: blue;">Hello</p>
```

✔ **2. Internal CSS**

Written inside <style> tag in <head>.

```
<style>
p { color: green; }
</style>
```

✔ **3. External CSS**

Written in a separate .css file.

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

3. What is the use of the “link” tag in HTML?

<link> tag is used to connect an external CSS file to an HTML document.

Example:

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

4. Define class selector and ID selector with examples.

✔ **Class Selector**

Used to style multiple elements.

```
.myClass {
  color: red;
}
```

HTML:

```
<p class="myClass">Text</p>
```

✔ **ID Selector**

Used for a **unique** element.

```
#myId {
  font-size: 20px;
}
```

HTML:

```
<p id="myId">Content</p>
```

5. What is the difference between internal and external CSS?

Internal CSS	External CSS
Written inside <style> tag	Written in .css file
Affects only that page	Can style entire website
Harder to maintain	Easier to maintain
Slows page if too large	Improves performance

6. Write the syntax to apply CSS to a paragraph element.

```
p {
  color: blue;
  font-size: 18px;
}
```

7. What is the purpose of the box model in CSS?

The **CSS box model** describes how every HTML element appears as a box.
It consists of:

- Content
- Padding
- Border
- Margin

It helps control spacing, layout, and alignment.

8. Define margin and padding in CSS.

✔ **Margin**

- Space **outside** the border
- Controls the gap between elements

✔ **Padding**

- Space **inside** the border
- Controls the space around content

Example:

```
margin: 20px;
padding: 10px;
```

9. How does the “float” property work in CSS?

float is used to position elements **left or right**, allowing text or images to wrap around them.

Example:

```
img {
  float: left;
}
```

10. What is the use of the “clear” property?

clear prevents elements from wrapping around floated elements.

Example:

```
div {
  clear: both;
}
```

11. Define “z-index” and write its purpose.

z-index controls the **stacking order** of overlapping elements.

Higher value = element appears on top.

Example:

```
div {
  z-index: 10;
  position: absolute;
}
```

12. How can CSS improve the appearance of a website?

CSS improves a website by:

- Adding colors, fonts, spacing
- Creating layouts
- Enhancing visual appeal
- Making pages responsive
- Maintaining consistency
- Reducing HTML code clutter

13. What is the role of the "default-router" property in network configuration?

➡ **Not related to CSS. Ignore.**

14. How can you assign different padding values for top, right, bottom, and left?

Using **padding shorthand:**

```
padding: 10px 20px 30px 40px;
```

This means:

- Top: 10px
- Right: 20px
- Bottom: 30px
- Left: 40px

15. Mention two benefits of using external CSS.

1. Same CSS file can style **multiple webpages**
2. Website becomes lightweight and easier to maintain

16. Write the command to link an external CSS file.

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

17. Explain how CSS helps in creating responsive web pages.

CSS supports:

- **Media queries**
- **Flexible layouts**
- **Percentage-based width**
- **Responsive images**

Example (media query):

```
@media (max-width:600px) {
  div { width: 100%; }
}
```

This makes websites adapt to mobile, tablet, and desktop screens.

18. What are pseudo-classes in CSS?

Pseudo-classes define a special state of an element.

Examples:

```
a:hover   → when mouse is over
a:visited → when link is visited
input:focus → when field is active
```

19. How can CSS be used to style lists or links?

✔ **Styling Lists:**

```
ul {
  list-style-type: square;
}
```

✔ **Styling Links:**

```
a {
  color: red;
  text-decoration: none;
}
a:hover {
  color: blue;
}
```

20. Explain how CSS helps in maintaining consistency across a website.

- A single external CSS file can style all pages
- Same colors, fonts, layout everywhere
- Makes website look professional
- Easy to update — change once, apply everywhere
- Reduces errors and mismatched designs

21. What is CSS? Explain its features and importance.

✔ **CSS (Cascading Style Sheets)**

CSS is a stylesheet language used to control the **design, layout, and appearance** of HTML pages.

✔ **Features:**

1. Controls colors, fonts, background, spacing
2. Helps create responsive web designs
3. Supports animations and transitions
4. Allows separation of content and design
5. Reusable code through external files

✔ **Importance:**

- Makes webpages attractive and consistent
- Saves time (change once → applied everywhere)
- Reduces HTML clutter
- Improves user experience and readability

22. Write the syntax of CSS declaration with example.

✔ **Syntax:**

```
selector {
  property: value;
}
```

✔ **Example:**

```
p {
  color: blue;
  font-size: 20px;
}
```

23. Explain different ways of using CSS in an HTML page.

✔ 1. Inline CSS

Written inside HTML tag.

```
<p style="color:red;">Hello</p>
```

✔ 2. Internal CSS

Inside <style> tag in <head>.

```
<style>
h1 { color: green; }
</style>
```

✔ 3. External CSS

Separate .css file linked to HTML.

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

24. What is an external CSS file? Write a sample CSS file.

An **external CSS file** is a separate .css file containing CSS rules.
It keeps HTML clean and allows reusability across multiple pages.

✔ Sample CSS file: style.css

```
body {
  background-color: lightgray;
}

h1 {
  color: blue;
}

p {
  font-size: 18px;
}
```

25. Define and explain selectors in CSS with examples.

Selectors are patterns used to select HTML elements for styling.

✔ **Types of Selectors:**

1. **Element Selector**

p { color: red; }

2. **Class Selector**

.myClass { font-size: 20px; }

3. **ID Selector**

#unique { background: yellow; }

4. **Universal Selector**

* { margin: 0; padding: 0; }

26. What is the CSS box model? Explain its components.

Every element in CSS is treated as a **box**, consisting of:

✔ **Components:**

1. **Content** → Actual text/image
2. **Padding** → Space inside the border
3. **Border** → Surrounds padding
4. **Margin** → Space outside the border

| Margin |
| Border |
| Padding |
| Content |

It helps control layout, spacing, and alignment.

27. What is padding in CSS? How can it be applied?

Padding is the **space between the content and the border** of an element.

✔ **Apply Padding:**

padding: 20px;

✔ **Different Padding Values:**

padding: 10px 15px 20px 25px;

(top, right, bottom, left)

28. Explain float and clear properties with examples.

✔ **float**

Positions an element left or right.

```
img {
  float: left;
}
```

✔ **clear**

Prevents elements from wrapping around floated items.

```
div {
  clear: both;
}
```

29. What is z-index in CSS? How does it work?

z-index controls **which element appears on top** when elements overlap.

✔ **Works only with positioned elements:**

(position: relative, absolute, fixed)

✔ **Example:**

```
div {
  position: absolute;
  z-index: 10;
}
```

Higher z-index = appears above others.

30. Write the benefits of using CSS in websites.

1. Better design and visual appeal
2. Faster page load time
3. Consistency across all pages
4. Easy maintenance
5. Reusable code
6. Creates responsive layouts

31. Explain how CSS helps in faster page loading and better design.

✔ **Faster Page Loading**

- External CSS loads once and is cached
- Reduces HTML file size
- Less inline styling → smaller code

✔ **Better Design**

- Advanced styling (colors, fonts, animations)
- Layout control via box model
- Responsive design using media queries

32. Differentiate between class selector and ID selector in CSS.

Class Selector	**ID Selector**
Used for multiple elements	Used for unique element
Starts with .	Starts with #
Less priority	Higher priority
Example: .item	Example: #header

33. How does padding affect the layout of an element?

Padding:

- Increases space **inside** the element
- Makes elements bigger
- Affects height/width calculations
- Pushes content away from borders

Example:

padding: 20px;

→ Add 20px inside border

34. Describe how to control the stacking order of elements using CSS.

Use **z-index** with positioning.

```
.box1 {
  position: absolute;
  z-index: 5;
}

.box2 {
  position: absolute;
  z-index: 10;
}
```

Element with **z-index 10** will appear on top.

35. Explain the importance of separating content and design using CSS.

Separating content (HTML) and design (CSS):

- Makes code clean and understandable
- Easy to update design without touching HTML
- Improves website performance
- Better SEO
- Allows consistent styling across all pages
- Simplifies maintenance for large websites

CHAPTER-IV CLIENT SIDE SCRIPTING

1. Explain Client-side scripting VS Server-side scripting with examples.

✔ **Client-side Scripting**

- Runs in the **browser**
- Code is visible to the user
- Faster execution
- Used for UI interactions, validation
- Example languages: **JavaScript**, HTML, CSS

Example:
JavaScript form validation in browser.

✔ **Server-side Scripting**

- Runs on **server**
- User cannot see code
- Used for database operations, login, backend logic
- Example languages: **PHP, Python, Node.js, Java**

Example:
Server checks login credentials from database.

2. Explain JavaScript basic syntax rules with suitable examples.

1. Statements end with **semicolon**
   let x = 10;
2. Case-sensitive
   Name ≠ name
3. Curly braces for blocks
4. if (x > 10) {
5.   console.log("OK");
6. }
7. Comments
   // single-line
   /* multi-line */

3. What are variables & identifiers? Explain naming rules with examples.

✔ **Variable**

Used to store data.

let age = 20;

✔ **Identifier**

Name given to a variable, function, etc.

✔ **Naming Rules**

- Cannot start with number → ✖ 1name
- Cannot use space → ✖ full name
- Only _ or $ allowed at start
- Case-sensitive
- Cannot use reserved words (like function, var)

Correct examples:

let myName;
let $price;
let _count;

4. Write a note on JavaScript data types & values with examples.

✔ **Primitive Data Types**

1. **Number**
   let x = 10;
2. **String**
   "Hello"
3. **Boolean**
   true, false
4. **Undefined**
   variable declared but no value
5. **Null**
   empty value
6. **Symbol**

7. **BigInt**

✔ **Non-Primitive Data Types**

- **Object**, **Array**, **Function**

Example:

```
let student = { name: "Atul", age: 20 };
let arr = [1,2,3];
```

5. Explain Scope in JavaScript (global vs function vs block scope) with example.

✔ **Global Scope**

Accessible everywhere.

```
var a = 10; // global
```

✔ **Function Scope**

Variables inside a function.

```
function test() {
  let x = 5; // function scope
}
```

✔ **Block Scope**

Variables inside { }
Only with **let** and **const**.

```
{
  let y = 20;
}
```

6. What are Literals in JavaScript? Explain its types with examples.

Literals = fixed values in code.

Types:

- **Numeric literal:** 100
- **String literal:** "Atul"
- **Boolean literal:** true

- **Array literal:** [1,2,3]
- **Object literal:** {name:"Aanya", age:17}
- **Null literal:** null

7. Write short note on Reserved Words in JavaScript.

Reserved words have special meaning in JS.
Cannot be used as variable names.

Examples:
var, let, const, if, else, switch, function, return, for, while, class, new

8. Explain Operators in JavaScript with examples.

✔ **Arithmetic Operators**

+ - * / % **
Example: 10 + 5

✔ **Relational Operators**

> < >= <= == !=
Example: x > y

✔ **Logical Operators**

&& || !
Example: (age > 18 && age < 60)

✔ **Assignment Operators**

= += -= *= /=
Example: x += 5;

9. Explain Control Statements in JavaScript (if, switch, loops) with examples.

✔ **if Statement**

```
if (age > 18) {
  console.log("Adult");
}
```

✔ **switch Statement**

```
switch(day) {
  case 1: console.log("Mon"); break;
}
```

✔ **Loops**

- **for loop**

```
for(let i=0;i<5;i++) console.log(i);
```

- **while loop**

```
while(x<5) x++;
```

- **do-while loop**

10. Explain Functions in JavaScript (User-defined, Built-in). Also explain parameters & return.

✔ **User-defined Function**

```
function greet(name) {
  return "Hello " + name;
}
```

✔ **Built-in Functions**

parseInt(), alert(), Math.random()

✔ **Parameters**

Values passed into a function.

✔ **Return**

Sends value back.

11. Explain JavaScript Objects. Explain Math / String / Date Object with methods.

✔ **JavaScript Object**

Collection of key-value pairs.

```
let person = {name:"Atul", age:20};
```

✔ Math Object

```
Math.sqrt(25)
Math.random()
Math.floor(4.8)
```

✔ String Object

```
"Atul".toUpperCase()
"Hello".length
"Hi there".includes("Hi")
```

✔ Date Object

```
let d = new Date();
d.getFullYear();
d.getMonth();
d.getDate();
```

12. Write short note on Regular Expressions with examples.

Regular expressions (RegEx) are patterns to match text.

✔ Example:

Check if email is valid:

```
let re = /^[a-z0-9]+@[a-z]+\.[a-z]{2,3}$/;
```

Match digits:

```
/[0-9]+/
```

13. Explain DOM in JavaScript. How to access & change HTML elements using DOM?

DOM (Document Object Model) represents HTML as a tree.

✔ Access Elements:

```
document.getElementById("title")
document.querySelector(".class")
```

✔ Change Content:

```
document.getElementById("title").innerHTML = "Welcome!";
```

✔ **Change Style:**

```
document.getElementById("title").style.color = "red"
```

14. Explain Event Handling in JavaScript with example.

Events = actions by user.

✔ **onclick()**

```
<button onclick="alert('Clicked!')">Click Me</button>
```

✔ **onmouseover()**

```
<p onmouseover="this.style.color='red'">Hover me</p>
```

✔ **onsubmit()**

```
<form onsubmit="return check()">...</form>
```

15. Write JavaScript code to validate an HTML form (email / mobile / password).

✔ **Example Validation:**

```
function validate() {
  let email = document.getElementById("email").value;
  let mobile = document.getElementById("mobile").value;

  let emailRe = /^[a-z0-9]+@[a-z]+\.[a-z]{2,3}$/;
  let mobRe = /^[0-9]{10}$/;

  if(!emailRe.test(email)) {
    alert("Invalid Email");
    return false;
  }

  if(!mobRe.test(mobile)) {
    alert("Invalid Mobile");
    return false;
  }
  return true;
}
```

16. Explain JavaScript variables, data types and values with examples.

✔ Variables

Variables are containers used to store data values in JavaScript.

Example:

```
let name = "Atul";
var age = 20;
const PI = 3.14;
```

✔ Data Types

JavaScript supports **two categories**:

◈ *Primitive Data Types*

1. **Number** → let x = 50;
2. **String** → "Hello"
3. **Boolean** → true / false
4. **Undefined** → declared but no value
5. **Null** → empty value
6. **Symbol**
7. **BigInt**

◈ *Non-Primitive (Reference) Types*

1. **Object** → let student = {name:"Aanya", age:17};
2. **Array** → [1,2,3]
3. **Function**

✔ Values

Values are actual data stored inside variables.

Example:

```
let marks = 90;       // Number
let msg = "Welcome";  // String
let active = true;    // Boolean
```

17. Explain scope in JavaScript (global, local, block) with examples.

✔ **1. Global Scope**

Variables accessible **throughout the entire program**.

```
var x = 10;   // global

function show() {
  console.log(x);   // accessible
}
```

✔ **2. Local Scope (Function Scope)**

Variables declared **inside a function**.

```
function test() {
  let a = 5;  // local
}
console.log(a); // ✕ Error
```

✔ **3. Block Scope**

Available only inside { }
Uses **let** or **const**.

```
{
  let y = 20;
}
console.log(y); // ✕ Error
```

18. Explain JavaScript operators (arithmetic, relational, logical, assignment) with examples.

✔ **1. Arithmetic Operators**

```
+ - * / % **
```

```
let a = 10 + 5;   // 15
```

✔ **2. Relational (Comparison) Operators**

```
> < >= <= == === !=
```

10 > 5   // true

✔ 3. Logical Operators

&& || !

(a > 5 && a < 20)

✔ 4. Assignment Operators

= += -= *= /= %=

```
let x = 10;
x += 5;   // 15
```

19. Explain DOM & Event Handling in JavaScript with suitable example.

✔ DOM (Document Object Model)

DOM represents the HTML page as a **tree of elements**.
Using JavaScript, we can **access and modify** HTML.

◈ Accessing Elements

```
document.getElementById("title")
document.querySelector(".box")
```

◈ Changing HTML Content

```
document.getElementById("title").innerHTML = "Hello Atul!";
```

✔ Event Handling

Event handling means performing actions when a user interacts with an element.

◈ Example: onclick Event

```
<button onclick="msg()">Click Me</button>

<script>
function msg() {
  alert("Button Clicked!");
}
</script>
```

💠 onmouseover Event

```
<p onmouseover="this.style.color='red'">Hover over me</p>
```

💠 onsubmit Event

```
<form onsubmit="return check()">
```

20. Explain JavaScript Objects. Also explain Math, String and Date objects with methods.

✔ JavaScript Object

An object is a collection of **key–value pairs**.

```
let person = {
  name: "Atul",
  age: 20
};
```

☆ SHORT QUESTIONS (2 Marks / MCQ Type)

1. What is difference between let, var, const?

var	let	const
function scoped	block scoped	block scoped
can be redeclared	cannot be redeclared	cannot be redeclared
value change allowed	allowed	not allowed

2. What is NaN?

NaN = Not a Number
Represents an invalid number value.
Example: "abc" / 2 → NaN

3. What is typeof operator?

Used to check data type of value.

typeof 10 → "number"

typeof "hi" → "string"

4. What is callback function?

A function passed as **argument** to another function.

Example:

setTimeout(function(){ alert("Hello!"); }, 1000);

5. Use of strict equal (===).

=== checks **value + data type both**.

```
5 === "5"   // false
5 == "5"    // true
```

CHAPTER-V (JSON)

1. Explain JSON overview + why JSON is widely used in web applications? Compare JSON with XML.

✔ JSON (JavaScript Object Notation)

JSON is a lightweight data format used for storing and exchanging data between a client and a server.
It is text-based, human-readable, and easy for machines to parse.

✔ Why JSON is widely used?

- Very easy to read/write
- Faster than XML
- Uses less data (lightweight)
- Supported by all programming languages
- Works naturally with JavaScript
- Ideal for APIs and web services

☆ JSON vs XML

JSON	XML
Lightweight	Heavier
Uses key–value pairs	Uses tags

JSON	XML
Easy to read/write	More complex
Faster parsing	Slower parsing
No end tags	Uses opening + closing tags
Preferred in modern APIs	Older web systems

Example JSON:

{"name": "Atul", "age": 20}

Example XML:

<student><name>Atul</name><age>20</age></student>

2. Write JSON syntax rules. Explain proper key/value structure with example. Also write what is allowed & what is not allowed in JSON.

✔ JSON Syntax Rules

1. Data is in **key–value pairs**
2. Keys must be **strings in double quotes**
3. Values can be string, number, boolean, null, array, object
4. Data is separated by **commas**
5. Object is inside { }
6. Array is inside [ ]

✔ Proper Key–Value Example

```
{
 "name": "Atul",
 "age": 20,
 "isStudent": true
}
```

✔ Allowed in JSON

- Strings in **double quotes**
- Numbers
- true / false
- null

- Arrays
- Objects

❌ **NOT ALLOWED in JSON**

- Single quotes → 'Atul' ❌
- Trailing commas → "age":20, } ❌
- Comments (// or /**/) ❌
- Functions ❌

3. Explain JSON data types (String, Number, Boolean, Null, Object, Array) with examples.

✔ **1. String**

"name": "Atul"

✔ **2. Number**

"marks": 95

✔ **3. Boolean**

"isStudent": true

✔ **4. Null**

"middleName": null

✔ **5. Object**

"address": {"city": "Delhi", "pin": 110001}

✔ **6. Array**

"skills": ["HTML", "CSS", "JS"]

4. What is a JSON object? Explain objects vs arrays in JSON with examples.

✔ **JSON Object**

A collection of **key–value pairs** inside { }.

Example:

```
{
 "name": "Atul",
 "age": 20
}
```

✔ **Objects vs Arrays**

JSON Object	JSON Array
{ }	[ ]
Key-value pairs	Ordered list
Represents an entity	Represents a list
Example: student	Example: list of students

✔ **Examples:**

Object:

```
{"name": "Aanya", "age": 17}
```

Array:

```
["Aanya", "Atul", "Riya"]
```

Array of Objects:

```
[
  {"name":"Aanya", "age":17},
  {"name":"Atul", "age":20}
]
```

5. Explain JSON Schema. Why schema validation is needed? Write sample JSON schema for student record.

✔ **JSON Schema**

A JSON schema defines the **structure**, **data types**, and **rules** for a JSON document.

It ensures the data is:

- Correct
- Valid
- In proper format
- Safe

✔ Why Validation Needed?

- Prevents wrong data
- Ensures API receives expected format
- Helps debugging
- Avoids runtime errors

✔ Sample JSON Schema for Student Record

```
{
  "type": "object",
  "properties": {
    "name": {"type": "string"},
    "age": {"type": "number"},
    "roll": {"type": "number"},
    "isStudent": {"type": "boolean"}
  },
  "required": ["name", "age", "roll"]
}
```

6. How to convert (serialize) JavaScript object into JSON string? Explain JSON.stringify() with example.

✔ JSON.stringify()

Converts a JavaScript object → JSON string (serialization).

✔ Example:

```
let student = {
  name: "Atul",
  age: 20
};

let jsonText = JSON.stringify(student);
console.log(jsonText);
```

Output:

```
{"name":"Atul","age":20}
```

7. How to convert JSON string into JavaScript object? Explain JSON.parse() with example + error conditions.

✔ **JSON.parse()**

Converts JSON string → JavaScript object.

✔ **Example:**

```
let txt = '{"name":"Aanya","age":17}';
let obj = JSON.parse(txt);

console.log(obj.name);
```

✔ **Error Conditions**

- If JSON has **single quotes**, parse fails
- If JSON has trailing commas
- If JSON contains comments
- If JSON syntax is invalid

8. Explain JSON Object in detail with example.

A JSON Object is a set of **key-value pairs** enclosed in **{ }**.

✔ **Example:**

```
{
 "id": 101,
 "name": "Atul",
 "skills": ["HTML", "CSS", "JS"],
 "address": {
   "city": "Delhi",
   "pincode": 110001
 }
}
```

This object contains:

- number
- string
- array

- another object

9. Explain data type of JSON in detail.

JSON supports **6 basic data types**:

✔ 1. String

Text inside double quotes
"hello"

✔ 2. Number

No separate int/float
10, 20.5

✔ 3. Boolean

true or false

✔ 4. Null

Represents empty
null

✔ 5. Object

Key-value pairs
{"name":"Atul"}

✔ 6. Array

Ordered list
["pen","book"]