CSS Theory Questions with Answers

Q: Role of CSS in separating structure and presentation.

A: CSS (Cascading Style Sheets) allows web developers to separate content from design. HTML provides the structure (e.g., headings, paragraphs, images), while CSS defines how elements appear (colors, fonts, spacing, layout). This separation improves maintainability, reusability, and consistency across web pages.

Q: Normal flow and positioning in CSS.

A: Normal flow is the default layout behavior of elements: block-level elements appear one after another vertically, and inline elements flow horizontally within their container. CSS positioning modifies this flow:

- static: default positioning.
- relative: positioned relative to its normal position.
- absolute: positioned relative to the nearest positioned ancestor.
- fixed: positioned relative to the viewport.
- sticky: switches between relative and fixed depending on scroll.

Q: Box model in CSS with figure. Margin vs Padding.

A: CSS box model consists of:

[Margin Border Padding Content]

- Margin: space outside the element's border.
- Padding: space between the content and the border.

This model determines how elements occupy space on a page.

Q: Inline vs block vs inline-block elements.

A: Inline: Flows with text, cannot set width/height (e.g., , <a>)

Block: Starts on a new line, takes full width (e.g., , <p>)

Inline-block: Flows like inline but allows width/height (e.g., <button>, <input>)

Q: Create a design (specify or Ill generate one).

A: Example design: A centered card with a title and description.

HTML:

```
<div class='card'>
  <h2>Welcome</h2>
  <p>This is a card layout.</p>
</div>
```

CSS:

```
.card {
  width: 300px; margin: 50px auto; padding: 20px;
  background: #f9f9f9; box-shadow: 0 0 10px #ccc;
  text-align: center; border-radius: 10px;
}
```

Q: CSS units Absolute vs relative (px, %, em, rem) with examples.

A: - px: absolute unit (e.g., font-size: 16px)

- %: relative to parent (e.g., width: 50%)

- em: relative to parent font-size (e.g., padding: 2em)

- rem: relative to root font-size (e.g., font-size: 1.5rem)

Q: ID vs Class selectors.

A: ID selector: #idName must be unique per page (e.g., #header)

Class selector: .className reusable on multiple elements (e.g., .card)

Use ID for unique elements, class for reusable styling.

Q: CSS position: static, relative, absolute, sticky, fixed (with examples).

A: - static: default (e.g., position: static;)

- relative: offset from normal (e.g., position: relative; top: 10px;)

- absolute: positioned to nearest ancestor (e.g., position: absolute; right: 0;)

- fixed: stays in place (e.g., position: fixed; top: 0;)

- sticky: scrolls until a point (e.g., position: sticky; top: 20px;)

Q: CSS specificity: how it influences rule application.

A: Specificity determines which CSS rule applies when multiple rules target the same element.

Priority order:

1. !important

2. Inline styles (style="")

3. ID selectors (#id)

4. Class selectors (.class)

5. Element selectors (div, p)

More specific rules override less specific ones.

HTML Theory Questions with Answers

Q: What is a tag in HTML? Explain structure, formatting, list, hyperlink, and executable tags.

A: A tag in HTML defines elements in a web page. Tags are enclosed in angle brackets like <tag>.

Structure tags include <html>, <head>, <body>.

Formatting tags include <b>, <i>, <p>.

List tags are <ul>, <ol>, <li>.

Hyperlink tag is <a href="...">.

Executable tags include <script> and <iframe>.

Q: Differences between inline and block-level elements (with 5 examples).

A: Inline elements do not start on a new line and only take up as much width as necessary. Examples: , <a>, <img>, , .

Block-level elements start on a new line and stretch to fill the container. Examples: , <p>, <h1>, <section>, <article>.

Q: How does a web server connect to the internet? How do we navigate URLs?

A: A web server connects via an ISP and is assigned a public IP. DNS translates domain names to IPs.

When a user enters a URL, the browser sends a request to the server using that IP, and the server responds with the requested page.

Q: How to map images in HTML? Hyperlinks with images and types.

A: <img src="image.jpg" alt="desc" usemap="#map">

```
<map name="map">
  <area shape="rect" coords="34,44,270,350" href="link.html">
</map>
```

Types: rectangular, circular, polygonal areas can be hyperlinked.

Q: Grouping of elements in HTML - types of grouping tags.

A: Grouping tags are used to organize content. Common ones:

 - Block-level container.

 - Inline container.

<section>, <article>, <nav>, <aside> - Semantic grouping tags.

Q: How to introduce multimedia in HTML? Explain with example.

A: Use <video> and <audio> tags.

Example:

```
<video controls>
  <source src="video.mp4" type="video/mp4">
</video>
<audio controls>
  <source src="audio.mp3" type="audio/mpeg">
</audio>
```

Q: Attributes of <form> tag. Differences between GET and POST.

A: Attributes: action (URL), method (GET/POST), enctype (for file upload).

GET: Appends data to URL, visible, limited length.

POST: Sends data in body, hidden, secure, no length limit.

Q: Difference between client-side and server-side rendering.

A: Client-side rendering: Rendered in browser using JS frameworks (React, Angular). Fast interactions but slower initial load.

Server-side rendering: Rendered on server (PHP, Django), better SEO and faster initial load.

Q: Difference between and . When to use each.

A: is a block-level element used for layout and grouping large sections.

 is an inline element used to group text for styling within lines.

Q: Accessibility in HTML - Role and how to improve accessibility.

A: Role: Ensure web usability for people with disabilities.

Improvements: Use semantic tags, alt text for images, ARIA attributes, proper form labels, keyboard navigation support.

Q: Designing accessible forms - What HTML elements/attributes would you use?

A: Use <label for="id">, placeholder, required, aria-describedby, fieldset, and legend.

Ensure keyboard navigability and screen-reader friendliness.

Assignment 1 - Theory with Answers

1. What happens when you type a URL (like https://pu.edu.np)? Explain with diagram.

Step-by-Step Process:

1. URL Parsing - Extracts domain and protocol.

2. DNS Lookup - Converts domain to IP.

3. TCP Handshake - Establishes connection (SYN -> SYN-ACK -> ACK).

4. TLS Handshake - Encrypts communication.

5. HTTP Request - Browser sends GET request.

6. Server Processing - Server processes and responds.

7. HTTP Response - Browser receives resources.

8. Rendering - Browser renders page.

Diagram:

[User] -> [DNS] -> [TCP] -> [TLS] -> [Server] -> [Response] -> [Rendering]

2. Explain any 5 commonly used protocols.

- HTTP: Transfers web data.

- HTTPS: Secure HTTP with encryption.

- FTP: Transfers files.

- SMTP: Sends emails.

- TCP/IP: Core Internet communication protocol.

3. Short notes on a Fully Qualified Domain Name (FQDN) with examples.

An FQDN is the full address of a website, including the hostname and domain.

Example: mail.google.com - 'mail' is the host, 'google.com' is the domain.

4. What is tier technology? Explain 1, 2, 3, and N-tier.

- 1-Tier: All layers in one system (monolithic).

- 2-Tier: Client (UI) and Server (DB).

- 3-Tier: UI + Logic + DB.

- N-Tier: Multiple layers with scalability.

5. Difference between HTTP and HTTPS. Why HTTPS is important today?

HTTP: Unsecured, uses port 80.

HTTPS: Encrypted (TLS), uses port 443.

Importance: Data security, SEO boost, user trust.

6. Explain how email is sent and received between two people.

1. Sender composes and sends email (via SMTP).

2. Mail server forwards to recipient server.

3. Recipient server stores email (IMAP/POP3).

4. Recipient accesses email via client.

7. What is a web application? Characteristics, needs, pros/cons, and best practices.

- Web App: Software accessed via browser.

- Characteristics: Dynamic, scalable, cross-platform.

- Needs: UI, backend, DB.

- Pros: Easy access/updates. Cons: Security, performance.

- Best Practices: Secure code, responsive UI, SEO.

8. As a web developer at ABC Company, explain setting up email, DNS, domain, hosting, SEO, and migration.

- Email: Configure via SMTP (e.g., Gmail).

- DNS: Use A/CNAME records.

- Domain: Register through providers.

- Hosting: Shared/VPS/cloud.

- SEO: Optimize tags, speed, mobile.

- Migration: Backup, transfer files, update DNS.

9. Difference between static/dynamic websites and front-end/back-end.

- Static: Fixed content, no DB.

- Dynamic: Server-side generated, uses DB.

- Front-end: UI (HTML, CSS, JS).

- Back-end: Server logic, DB, APIs.

10. Break down this sample URL: https://www.ncit.edu.np/store/books?category=fiction#top

- Protocol: https

- Subdomain: www

- Domain: ncit.edu.np

- Path: /store/books

- Query: ?category=fiction

- Fragment: #top