1) <!DOCTYPE html> Is it a tag of HTML? If not, what is it and why do we use it?

Ans:

1. `<!DOCTYPE html>` is not an HTML tag; it is a Document Type Declaration (DTD).

2. It is used to specify the version of HTML or XML being used in a web page.

3. The declaration `<!DOCTYPE html>` specifically indicates that the web page is written in HTML5.

4. The doctype declaration is placed at the very beginning of an HTML document, before the `<html>` tag.

5. It is not an HTML tag itself but a directive that instructs the web browser on how to interpret and render the rest of the HTML code.

6. Its purpose is to ensure that the web page is rendered correctly by web browsers.

7. Different versions of HTML have different rules and syntax, so the doctype declaration informs the browser which set of rules to follow.

8. The `<!DOCTYPE html>` declaration triggers the browser's "standards mode" and provides better compatibility across different browsers.

9. HTML5 introduced a simplified and concise doctype declaration compared to previous HTML versions.

10. Including the correct doctype declaration is essential for proper rendering and compatibility with modern web browsers.

Q.2 Explain Semantic tags in HTML. And why do we need it?

Ans:

Semantic tags in HTML are elements that provide meaning and structure to the content within a web page. They convey the purpose or role of the content to both browsers and developers. Semantic tags describe the meaning of the content rather than just specifying how it should be displayed.

Few examples of semantic tags introduced in HTML5:

1. `<header>`: Represents the introductory content or a container for a group of introductory or navigational elements.

2. `<nav>`: Defines a section of navigation links.

3. `<main>`: Specifies the main content of a document, excluding headers, footers, and sidebars.

4. `<article>`: Represents a self-contained composition, such as a blog post, article, or news story.

5. `<section>`: Defines a standalone section of content within a document.

6. `<aside>`: Represents content that is tangentially related to the surrounding content, such as sidebars or pull quotes.

7. `<footer>`: Represents the footer of a document or a section.

The use of semantic tags is beneficial for several reasons:

1. **Accessibility**: Semantic tags help assistive technologies, such as screen readers, to understand and navigate the content more accurately. This improves the accessibility of the website for users with disabilities.

2. **Search Engine Optimization (SEO)**: Search engines rely on the structure and semantics of HTML to understand the content of web pages. By using semantic tags, you can provide clear signals to search engines about the purpose and hierarchy of the content, potentially improving your website's visibility in search results.

3. **Code Readability and Maintainability**: Semantic tags make the HTML code more meaningful and easier to understand for developers. By using tags that accurately describe the content, it becomes simpler to navigate, modify, and maintain the codebase, especially in larger projects.

4. **Future-proofing**: Semantic tags make your HTML more forward-compatible. As web standards evolve, browsers and technologies can better adapt to new features and functionalities if the content is marked up semantically.

5. **Styling and Consistency**: Semantic tags allow developers and designers to apply consistent styling to similar types of content. By selecting semantic tags instead of generic ones (e.g., `<div>` or `<span>`), it becomes easier to target and style specific sections of a web page.

Q.3 Differentiate between HTML Tags and Elements?

Ans:

HTML Tags:

- HTML tags are the building blocks of HTML markup.

- They are used to define the structure and presentation of content within an HTML document.

- Tags are represented by opening and closing angle brackets (`<` and `>`).

- Examples of HTML tags include `<h1>`, `<p>`, `<div>`, `<a>`, etc.

- Tags can have attributes that provide additional information or modify their behavior.

- Tags can be nested inside each other to create a hierarchical structure.

HTML Elements:

- HTML elements are formed by combining HTML tags, along with their content, to create a complete unit.

- An HTML element consists of an opening tag, the content within it, and a closing tag.

- For example, `<p>` is an HTML tag, but `<p>Some text</p>` is an HTML element.

- Elements represent the actual components or objects on a web page.

- Elements can include text, images, links, lists, tables, forms, and more.

- HTML elements can also be self-closing, meaning they don't require a closing tag. For example, `<img>`.

Q.6 What are some of the advantages of HTML5 over its previous versions?

Ans:

The advantages of HTML5:

1. **Simplified Doctype**: HTML5 introduced a simplified and standardized doctype declaration (`<!DOCTYPE html>`), which is easier to remember and use.

2. **New Semantic Tags**: HTML5 introduced a set of semantic tags (e.g., `<header>`, `<nav>`, `<section>`, `<footer>`, etc.), making it easier to define the structure and purpose of different sections of a webpage. This improves accessibility, SEO, and code readability.

3. **Multimedia Support**: HTML5 provides native support for multimedia elements, such as `<audio>` and `<video>`. This eliminates the need for third-party plugins like Flash and allows for easier embedding and customization of multimedia content.

4. **Canvas**: HTML5 introduced the `<canvas>` element, which allows for dynamic rendering of graphics, animations, and interactive visualizations directly within the browser, without the need for plugins or external libraries.

5. **Improved Form Input Types and Validation**: HTML5 introduced new input types (e.g., `<date>`, `<email>`, `<range>`, `<color>`, etc.) and enhanced form validation capabilities. This

simplifies form handling and validation, reduces reliance on JavaScript, and provides a better user experience.

6. **Offline Web Applications**: HTML5 introduced technologies like the Application Cache and Local Storage, which enable web applications to work offline or with limited connectivity. This improves performance and user experience for users accessing web applications in offline or unreliable network conditions.

7. **Geolocation**: HTML5 provides native support for geolocation, allowing web applications to access a user's location information with their permission. This enables location-aware applications and services.

8. **Improved Performance and Efficiency**: HTML5 introduced several features and APIs that enhance performance, such as async and defer attributes for script loading, improved caching mechanisms, and the ability to load scripts in the background.

9. **Compatibility and Consistency**: HTML5 has better cross-browser compatibility compared to older versions of HTML, reducing the need for browser-specific workarounds and improving consistency in rendering and functionality across different browsers.

10. **Mobile and Responsive Design**: HTML5 includes features that support mobile devices and responsive design, such as the viewport meta tag, which allows web pages to adapt to different screen sizes and orientations.

Q.8 What is the difference between <figure> tag and <img> tag?

Ans:

The `<figure>` tag and the `<img>` tag serve different purposes and have distinct roles in HTML:

1. `<img>` tag:

   - The `<img>` tag is used to insert an image into an HTML document.

   - It is a self-closing tag, meaning it doesn't require a closing tag.

   - The `<img>` tag requires the `src` attribute to specify the image source (URL or file path) and the `alt` attribute to provide alternative text for accessibility and SEO purposes.

   - It represents a standalone image within the document and is typically used when there is no need for additional caption or related content.

2. `<figure>` tag:

   - The `<figure>` tag is used to group together a self-contained content block that is referenced from the main flow of the document.

   - It can be used to encapsulate different types of content, including images, illustrations, videos, audio, code snippets, etc.

   - The `<figure>` tag should be used when the content within it is referenced or related to the surrounding text and needs a caption or description.

   - It is often accompanied by the `<figcaption>` tag, which is used to provide a caption or description for the content within the `<figure>` element.

   - The `<figure>` tag provides semantic meaning and helps improve accessibility and structure of the document.

Q.9 What’s the difference between HTML tag and attribute and give example of some global attributes?

Ans:

The HTML tag and attribute are two distinct concepts in HTML:

1. HTML Tag:

   - An HTML tag represents an element or a structure within an HTML document.

   - Tags are used to define the structure and semantics of the content.

   - Examples of HTML tags include `<h1>`, `<p>`, `<div>`, `<a>`, etc.

   - Tags are enclosed in angle brackets (`<` and `>`).

   - Tags can have attributes to provide additional information or modify their behavior.

2. HTML Attribute:

   - An HTML attribute provides additional information or modifies the behavior of an HTML element.

   - Attributes are specified within the opening tag of an HTML element.

   - Attributes consist of a name-value pair.

   - Attributes provide various functionalities such as defining the source of an image, specifying link destinations, setting styles, enabling interactivity, etc.

Example of some global attributes (attributes that can be used with any HTML element):

1. `id` attribute: Provides a unique identifier for an element, which can be used for styling or JavaScript manipulation.

Example: `<div id="myElement">Content</div>`

2. `class` attribute: Assigns one or more class names to an element, allowing CSS and JavaScript to target and style multiple elements at once.

   Example: `<p class="highlight">Highlighted Text</p>`

3. `style` attribute: Defines inline CSS styles for an element, allowing you to apply specific styles directly to the element.

   Example: `<span style="color: blue; font-size: 14px;">Styled Text</span>`

4. `src` attribute: Specifies the source URL or file path of media elements like images, audio, or video.

   Example: `<img src="image.jpg" alt="Image">`

5. `href` attribute: Specifies the URL or destination of a hyperlink.

   Example: `<a href="https://www.example.com">Link</a>`