5. What is BOM? DOM? Diff. between BOM vs DOM?

In JavaScript, BOM stands for the "Browser Object Model."

It is a set of objects provided by web browsers to interact with and manipulate the browser window. The Browser Object Model allows you to control various aspects of the browser, such as the browser window, frames, history, location, and more.
Some common objects and properties within the Browser Object Model include:

window: The global object representing the browser window.
document: Represents the web page's document and its content.
location: Provides information about the current URL and allows you to navigate to other web pages.
history: Allows you to interact with the browser's history, enabling navigation forward and backward.
navigator: Provides information about the user's browser and operating system.
screen: Contains information about the user's screen, like screen dimensions.
alert(), confirm(), prompt(): Functions for displaying dialog boxes in the browser.

The Browser Object Model is different from the Document Object Model (DOM), which deals with the structure and content of web documents (HTML or XML). BOM, on the other hand, focuses on browser-specific functionality and interactions.

In JavaScript, the DOM stands for the "Document Object Model." The DOM is a programming interface for web documents, and it represents the structure and content of a web page, allowing you to interact with and manipulate that page dynamically.
Here are some key points about the DOM in JavaScript:

Document Structure: The DOM represents the web page as a tree-like structure, with elements like HTML tags, attributes, and text content forming nodes in the tree. These nodes are interconnected to represent the structure of the page.

Access and Manipulation: JavaScript can be used to access, modify, or manipulate elements and their attributes on the web page. You can change the content, style, and behavior of elements using JavaScript and the DOM.

Event Handling: The DOM allows you to set up event listeners to respond to user actions, such as clicks, keypresses, or mouse movements. This enables you to create interactive and responsive web pages.

Traversal: You can traverse the DOM tree to move from one element to another, whether it's going up to a parent element or down to child elements, siblings, or other related elements.
Dynamic Updates: JavaScript and the DOM allow you to create dynamic web applications where content and structure can change without requiring a full page refresh. This is often seen in modern web applications, such as single-page applications (SPAs).
The DOM is an essential part of web development, as it allows you to create dynamic, interactive, and responsive web pages and applications by manipulating the content and structure of HTML documents using JavaScript.

The main differences between BOM (Browser Object Model) and DOM (Document Object Model) in web development are as follows:

Scope and Purpose:BOM (Browser Object Model): BOM focuses on interacting with and controlling the browser itself, rather than the web page's content. It provides access to browser-specific features like the window, location, history, and navigator objects.
DOM (Document Object Model): DOM is concerned with representing the structure and content of web documents (HTML or XML). It provides a structured way to access and manipulate the elements on a web page.

Components: BOM includes objects like window, document, location, history, and navigator.

DOM includes elements like HTML tags, attributes, and text content, structured as a tree with nodes representing the web page's structure.

Functionality: BOM is responsible for browser-related actions, such as opening new windows or tabs, navigating to different URLs, and managing the browser's history.

DOM allows you to access and manipulate the content, style, and structure of web documents. It handles events and user interactions within the web page itself.

In summary, BOM deals with controlling the browser environment and its features, while DOM deals with interacting with and modifying the content and structure of web documents displayed within the browser.

6. How to use document.getElementsByTagName() method?

The document.getElementsByTagName() method returns all the element of specified tag name.

syntax:

```
document.getElementsByTagName("name")
```

eval():

In JavaScript, eval() is a global function that allows you to execute code dynamically by taking a string as an argument and interpreting it as JavaScript code. Here's how it works:

```
var x = 10;
var y = 20;
var code = "var result = x + y; result * 2;"; // This is a string containing JavaScript
code.
var dynamicResult = eval(code); // The eval() function interprets the code string.

console.log(dynamicResult); // This will output the result of the evaluated code.
```

In this example, eval() takes the code variable, which is a string containing JavaScript code, and executes it.
In the end, dynamicResult contains the result of the evaluated code.
However, it's essential to be cautious when using eval() because it can be a security risk. When executing code from user input or external sources, it can potentially introduce security vulnerabilities if not used carefully. In many cases, there are safer alternatives to achieve dynamic code execution without using eval(), such as using functions or JSON.parse() for structured data.