**TASK2**

1. **The Distinctions Between Document and Window Objects in JavaScript**

**Document Object:**

The document object represents a web page that is loaded in the browser. By accessing the document object, we can access the element in the HTML page. With the help of document objects, we can add dynamic content to our web page. The document object can be accessed with a window. Document or just document.

**Syntax:**

**document.property\_name;**

1. **Scope of Operation:**
   * Deals primarily with the content within an HTML document.
2. **Hierarchy:**
   * Nested within the Window object, representing the structure and elements of the page.
3. **Access to HTML Elements:**
   * Provides methods like **getElementById** or **getElementsByClassName** to access and manipulate HTML elements within the document.
4. **Lifecycle:**
   * Associated with the lifecycle of the HTML document. Created when the HTML is parsed and ceases to exist when the page is closed or unloaded.
5. **Content Manipulation:**
   * Used for manipulating the structure and content of the HTML document, including adding, modifying, or deleting elements.
6. **Event Handling:**
   * Manages events related to the content of the document, such as clicks, keypresses, or form submissions.
7. **Properties:**
   * Contains properties that represent the document's characteristics, like **title**, **URL**, and **head**.
8. **Methods:**
   * Provides methods for selecting and modifying elements within the document, like **createElement**, **appendChild**, and **querySelector**.
9. **DOM Representation:**
   * Represents the hierarchical structure of the HTML document as a tree-like structure known as the DOM.
10. **Child Objects:**
    * Contains various sub-objects representing specific parts of the document, such as **body**, **forms**, and **links**.

**Some methods of Document Objects**

• activeElement: It returns the currently active elements in the document.

• body: It returns the contents of the body element.

• anchors: It returns all <a> elements that have a name attribute.

• forms: It returns all the elements of the form.

• title: It returns the title element of the document.

• head: It returns the head element of the document.

• links: It returns all <area> and <a> elements that have a href attribute.

**Window Object**

The window object is the topmost object of the DOM hierarchy. It represents a browser window or frame that displays the contents of the webpage. Whenever a window appears on the screen to display the contents of the document, the window object is created.

**Syntax:**

**window.property\_name;**

1. **Scope of Operation:**
   * Represents the entire browser window and is responsible for managing the browser itself.
2. **Hierarchy:**
   * Acts as the global object and contains the Document object as one of its properties.
3. **Access to HTML Elements:**
   * Does not directly provide methods for accessing HTML elements but encompasses the Document object, which handles such interactions.
4. **Lifecycle:**
   * Persists as long as the browser window is open, representing the browser instance itself.
5. **Content Manipulation:**
   * Primarily used for controlling the overall behavior of the browser, such as opening new windows or navigating to different URLs.
6. **Event Handling:**
   * Handles events related to the browser window, like resizing, scrolling, or closing.
7. **Properties:**
   * Contains properties related to the browser window, such as **innerWidth**, **innerHeight**, and **location**.
8. **Methods:**
   * Offers methods for manipulating the browser window, including **open**, **close**, and **alert**.
9. **DOM Representation:**
   * Does not directly represent the DOM structure but encapsulates the Document object, which does.
10. **Child Objects:**
    * Has child objects like **history**, **navigator**, and **screen**, each providing information about different aspects of the browser environment.

**Some of window methods:**

* alert(): It is used to display an alert box. It displays a specified message along with an OK button and is generally used to make sure that the information comes through the user.
* setInterval(): It repeats a given function at every given time interval.
* setTimeout(): It executes a function, after waiting a specified number of milliseconds.
* stop(): It is used to stop the window from loading resources in the current browsing context.