1. What is the difference between **getElementById, getElementsByClassName , and querySelector / querySelectorAll**?

   Answer: Difference between getElementById, getElementsByClassName, and querySelector / querySelectorAll

   getElementById VS getElementsByClassName:

   getElementById() : Get an element according to a specific id name. Returns the single element with the given id.

   getElementsByClassName (): Returns multiple elements by a class name.

   querySelector VS querySelectorAll:

   querySelector() : Returns the first element that matches the CSS selector**.**
   querySelectorAll() : Returns a static NodeList of all elements that match the CSS selector.

2. How do you **create and insert a new element into the DOM**?
   Answer:
   Step-1: Create a new element into the DOM : Make a new element
   Const newDiv = document.createElement("div")

   Step 2: Add attributes: Give here text,classes, or an id before putting it in the page.

```
newDiv.textContent = "Hello World!";
newDiv.className = "greeting";
newDiv.style.color = "blue";
```

   Step 3: Insert a new element into the DOM : Use appendChild() to place it inside another element,like the body.

```
document.body.appendChild(newDiv);
```

3. What is **Event Bubbling** and how does it work?

   Answer:

   Event Bubbling: Event bubbling is a DOM mechanism where, when an event is triggered on an element, that event does not bubble up through the child to the parent, meaning that if a child element is clicked, that event will also trigger its parent element.

   Html:

```
<div id="parent">
  <button id="child">Click me</button>
</div>
```

   Js:

```
document.getElementById("parent").addEventListener("click", () => {
```

```
  console.log("Parent clicked");
});
document.getElementById("child").addEventListener("click", () => {
  console.log("Child clicked");
});
```

4. What is **Event Delegation** in JavaScript? Why is it useful?

   Answer:

   Event Delegation: Event delegation involves handling events on child elements by passing the event from the component. It is necessary when there are many child elements and it is not possible to provide a separate event handler for each.

   Usefulness: Event Delegation helps in memory optimization. Handling dynamically added elements. Improving performance (fewer listeners).

5. What is the difference between **preventDefault() and stopPropagation()** methods?

   Answer:

   Difference between preventDefault() and stopPropagation() methods:

   PreventDefault(): Prevents the default browser behaviour of an element. Blocks the browser's default action.

   stopPropagation(): Stops the event from bubbling up the DOM. Stops the event from traveling up the DOM.