JS TOPICS

1. Explain setTimout.

Ans:

- The function takes two parameters: a callback function and a delay value in milliseconds. The callback function represents the code we want to execute after the delay.
- When we call setTimeout, it registers the callback function and starts a timer. After the specified delay, the JavaScript engine adds the callback function to the event queue.
- In case you don't know what an event Queue is , the event queue is a data structure that holds tasks to be processed by the JavaScript runtime. When the call stack is empty (all synchronous code has finished executing), the runtime picks the next task from the event queue and executes it.
- By using setTimeout, we introduce an asynchronous behavior in our code. This means that while the delay is counting down, the JavaScript engine can continue executing other code without waiting for the setTimeout callback to be invoked.

2. Explain cancelTimeOut

The clearTimeout function is used to cancel a timeout set by the setTimeout function. When you call clearTimeout with the identifier returned by setTimeout, it prevents the registered callback function from being executed if the timeout has not already occurred.

- If you call clearTimeout before the timeout has occurred (i.e., before the callback function is added to the event queue), the timeout is canceled, and the callback function will not be executed. If the timeout has already occurred and the callback function is in the event queue, calling clearTimeout will have no effect, as the callback will be executed from the event queue.

- In summary, if you call clearTimeout before the timeout triggers, it prevents the callback from being added to the event queue. If you call it after the timeout has triggered but before the callback is executed, it has no effect. Once the callback is

in the event queue or being executed, clearTimeout cannot stop it.