(Notes) Event Propagation

Question 1: Event Bubbling

Explanation:

Event bubbling refers to the propagation of an event from the innermost target element to the outermost ancestor. In the provided code, clicking on the button will also trigger the click event on the form and div elements due to event bubbling.

Question 2: event.target vs this.target vs event.currentTarget

Explanation:

- `event.target` refers to the element that triggered the event.
- `this` refers to the element to which the event listener is attached (i.e., the current context).
- `event.currentTarget` refers to the element that is currently handling the event during its capture or bubbling phase.

Question 3: Event Capturing

Explanation:

Event capturing is the opposite of event bubbling, where the event is captured from the outermost element towards the target element. By setting `{ capture: true }`, event listeners are triggered during the capturing phase instead of the bubbling phase.

Question 4: Stop Propagation

Explanation:

`event.stopPropagation()` prevents the further propagation of an event through the DOM tree. In the provided code, clicking on the form element will not trigger the click event on the div element due to `event.stopPropagation()` being called.

Question 5: Event Delegation

Explanation:

Event delegation is a technique where a parent element listens for events on behalf of its children. In this code, the parent element with the class `"products"` listens for click events and checks if the clicked target is a `span` element, redirecting to a specific URL based on the span's class.

Question 6: What is the Output?
Explanation:
The output depends on which element is clicked:

- If the **div** or **button** is clicked, it will alert `"div"` or `"button"` respectively.
- If the **form** is clicked, it will first trigger the capturing phase (alerting `"form"`) before the bubbling phase (alerting `"form"` again).

Question 7: Create a Modal which closes by clicking on negative space
Explanation:
This code sets up a modal that becomes visible when a button with the class `"modalButton"` is clicked. Clicking on the modal container (the area outside the modal content) will hide the modal by toggling its display property between `"flex"` and `"none"`.
