Module – 4 Lists and Hooks

1.Explain Life cycle in Class Component and functional component with Hooks

**=> I. Class Component Lifecycle:**

**1.Mounting Phase:**

* + **constructor()**: This is the first method called when a component is created. It's used for initializing state and binding event handlers.
  + **render()**: This method is responsible for returning the JSX that represents the UI of the component.
  + **componentDidMount()**: This method is invoked immediately after a component is mounted (inserted into the tree). It's commonly used to perform tasks such as data fetching or setting up subscriptions.

1. **Updating Phase:**
   * **shouldComponentUpdate(nextProps, nextState)**: This method allows you to control if the component should re-render when props or state change. It returns a boolean value indicating whether the component should update or not.
   * **render()**: This method is called again to re-render the component with updated props or state.
   * **componentDidUpdate(prevProps, prevState)**: This method is invoked immediately after updating occurs. It's often used to perform DOM operations or data fetching based on the updated state.
2. **Unmounting Phase:**
   * **componentWillUnmount()**: This method is called just before the component is removed from the DOM. It's used to perform cleanup tasks like unsubscribing from event listeners or canceling network requests.

**II. Functional Component with Hooks:**

1. **Mounting Phase:**
   * **useState()**: This hook allows functional components to manage state. It returns a stateful value and a function to update it, analogous to **this.state** and **this.setState()** in class components.
   * **useEffect(() => {}, [])**: This hook combines **componentDidMount**, **componentDidUpdate**, and **componentWillUnmount** lifecycle methods. It runs the provided function after every render, including the initial render. The second argument (an array) allows you to specify dependencies, and the effect will only re-run if one of these dependencies changes.
2. **Updating Phase:**
   * Same as mounting phase, **useState()** and **useEffect()** hooks are used for managing state and performing side effects, respectively. React will automatically re-run effects when certain dependencies change.
3. **Unmounting Phase:**
   * Similar to class components, cleanup in functional components with hooks can be done inside the cleanup function passed to **useEffect()**. If you return a function from the effect, React will call it when it's time to clean up.