**1. Explain the Need and Benefits of Component Life Cycle**  
In React, the component life cycle refers to the series of phases a component goes through from the moment it is created until it is removed from the DOM. Understanding the life cycle is important because it helps developers control what happens at specific points during a component's existence. For example, you might want to fetch data from an API when the component is first rendered, or clean up a timer or event listener just before it is removed. The main benefits of using life cycle methods include better control over component behavior, efficient resource management, improved performance, and the ability to respond to user interactions or external data updates in a structured way.

**2. Identify Various Life Cycle Hook Methods**  
React class components provide several lifecycle methods, which can be grouped based on the phase of the component: mounting, updating, and unmounting. During the mounting phase, the key methods are constructor(), getDerivedStateFromProps(), render(), and componentDidMount(). In the updating phase, commonly used methods include shouldComponentUpdate(), getSnapshotBeforeUpdate(), and componentDidUpdate(). Finally, during unmounting, componentWillUnmount() is used to clean up resources. In function components, the useEffect() **hook** can be used to mimic many of these lifecycle behaviors depending on how it's configured, making lifecycle management accessible without using class components.

**3. List the Sequence of Steps in Rendering a Component**  
When a React component is rendered, it follows a specific sequence of steps, especially in a **class component**. First, if it exists, the constructor() is called to initialize the state and bind methods. Then, getDerivedStateFromProps() may run to sync state with props if needed. Next, the render() method is called, which returns the JSX to be displayed. After the render, React updates the DOM, and then componentDidMount() is invoked, where you can perform actions like data fetching. During updates (when props or state change), the order is: getDerivedStateFromProps(), shouldComponentUpdate(), render(), getSnapshotBeforeUpdate(), and finally componentDidUpdate(). When the component is about to be removed from the DOM, componentWillUnmount() is called. This entire sequence ensures a controlled and optimized rendering process.