**Explain the need and Benefits of component life cycle**  
The component lifecycle in React provides a structured way to manage and respond to changes in a component’s state and behavior over time. Understanding the lifecycle is essential for performing tasks such as initializing data, making API calls, handling side effects, and cleaning up resources. The lifecycle methods allow developers to hook into specific phases of a component's existence—mounting, updating, and unmounting—enabling better control and predictability. This leads to more efficient, maintainable, and bug-free applications.

**Identify various life cycle hook methods**  
React class components include several lifecycle hook methods. Common ones include:

* constructor(): Initializes the component’s state and binds methods.
* componentDidMount(): Called after the component is inserted into the DOM; used for data fetching or DOM manipulation.
* shouldComponentUpdate(): Determines whether a re-render is necessary.
* componentDidUpdate(): Invoked after the component updates; used to act on prop or state changes.
* componentWillUnmount(): Called just before the component is removed from the DOM; used for cleanup tasks such as clearing timers or removing event listeners.

**List the sequence of steps in rendering a component**  
The typical sequence of steps when rendering a React class component is as follows:

1. The constructor() is called to initialize the component.
2. The render() method is invoked to return the JSX for the UI.
3. The component is mounted to the DOM.
4. The componentDidMount() method is called after the component is mounted.
5. If the component’s state or props change, shouldComponentUpdate() may be called to determine if a re-render is needed.
6. If the component updates, the render() method runs again.
7. After re-rendering, componentDidUpdate() is called.
8. When the component is about to be removed, componentWillUnmount() is executed.