**Module-10**

**1 : - Explain Life cycle in Class Component and functional component with Hooks**

**Ans : -**

- A React component undergoes three phases in its lifecycle: mounting, updating, and unmounting.

(1) The mounting phase is when a new component is created and inserted into the DOM or, in other words, when the life of a component begins. This can only happen once, and is often called “initial render.”

(2) The updating phase is when the component updates or re-renders. This reaction is triggered when the props are updated or when the state is updated. This phase can occur multiple times, which is kind of the point of React.

(3) The last phase within a component's lifecycle is the unmounting phase, when the component is removed from the DOM.

[**Life Cycle in Class Component**](https://github.com/Devanshu-73/TOPS_ASSIGNMENT/blob/master/Module%20-%20Lists%20and%20Hooks/Q-01.md#life-cycle-in-class-component-)**:-**

- Order of component mounting:

- constructor()

static getDerivedStateFromProps()

render()

componentDidMount()

- hope you noticed that componentDidMount is called just after the constructor invocation and the first render.

- It means that the component is already applied to the DOM and you can interact with it.

- componentDidMount is a good place to set a subscription, timeout, interval or invoke an API request**.**

### [Updating with componentDidUpdate](https://github.com/Devanshu-73/TOPS_ASSIGNMENT/blob/master/Module%20-%20Lists%20and%20Hooks/Q-01.md#updating-with-componentdidupdate) :-

- An update is caused by props or state changes. It goes with the following order

- static getDerivedStateFromProps()

- shouldComponentUpdate()

- render()

- getSnapshotBeforeUpdate()

- componentDidUpdate()

- componentDidUpdate happens just after the component re-render.

- This method is not called to the first render.

- You should use this method to make a request, set a state, or operate on the DOM.

- Please do not forget to compare current and previous props/state to avoid an infinite loop.

### [Unmounting the component](https://github.com/Devanshu-73/TOPS_ASSIGNMENT/blob/master/Module%20-%20Lists%20and%20Hooks/Q-01.md#unmounting-the-component) :-

- componentWillUnmount() is invoked just before a component is unmounted.

- You should use this method to clean up necessary timers, and subscriptions or cancel network requests.

## [Life Cycle in Functional Component :](https://github.com/Devanshu-73/TOPS_ASSIGNMENT/blob/master/Module%20-%20Lists%20and%20Hooks/Q-01.md#life-cycle-in-functional-component-)

(1) Mounting

(2) Updating

(3) Un-mounting

-Any react component undergoes these basic phases. Inserting components into the real DOM is called Mounting, Dynamically updating data inside components by rendering is called Updating, and removing components from the real DOM is called unmounting. Traditionally we have life cycle methods for class components functional components don't have these methods.

- But we can achieve these methods for functional components using hooks. Here comes the picture what are all hooks mean? Well, hooks are nothing but plain JavaScript functions which can add superpowers to our functional components. React developers already wrote some inbuilt super-powerful hooks for us. We can even create our own custom hooks and can use them throughout our app.

- By using these super powerful hooks provided by react we can assimilate class component behavior and functionality to our functional component with reduced complexity. Functional components are really simple and easy hands-on to get started.