# Q1: Explain Life cycle in Class Component and functional component with Hooks?

Ans:

Life cycle of class component in React

1. Constructor :-

* constructor(): Initializes state and binds methods.

1. ComponentDIdMount() :-

* ComponentDIdMount is a lifecycle method in react that is automatically called after the component has been mounted or rendered. It means that component is ready to run or component has taken birth.

1. ComponentDIdUpdate() :-

* ComponentDIdUpdate is a lifecycle method in react that is called immediately after a component has been updated or re-rendered.

1. ComponentWillUnmount() :-

* This method is called just before the component is removed from the DOM. It's typically used for cleaning up resources like event listeners.

# Example :-

# import React, { Component } from 'react';

# class ExampleClassComponent extends Component {

# componentDidMount() {

# // Fetch data or set up timers

# }

# componentDidUpdate(prevProps, prevState) {

# // Perform actions after an update

# }

# componentWillUnmount() {

# // Clean up resources }

# render() {

# return (<div>Class Component with Lifecycles</div>);

# }

## }

Lifecycle in Functional Components with Hooks :

1. Birth and Growth (Mounting and Updating) : -

* Functional components with hooks have two main phases :-
* useState : This is where you set up the creature's initial characteristics and state.
* useEffect : Similar to componentDidMount and componentDidUpdate in class components, this is where you can make your creature do things when it's born and whenever it changes. It's like a combination of those lifecycle methods.

1. Aging and Farewell (Unmounting): When the creature is no longer needed, you can use:

* useEffect cleanup: This is where you can clean up resources, similar to componentWillUnmount in class components.

Example:

import React, { useState, useEffect } from 'react';

const ExampleComponent = () => {

const [data, setData] = useState([]);

useEffect(() => {

// This runs after the component renders

fetchData();

return () => {

};

}, []); // Empty dependency array means this effect runs only once on mount

return (

<div>

{/\* Component rendering logic \*/}

</div>

);

}