**Question 1: What are Components in React? Explain the Difference Between Functional and Class Components.**

**Components** are reusable building blocks in React that define parts of the UI. They help create **modular, maintainable, and scalable** applications.

**Types of Components**

1. **Functional Components**
   * Simple JavaScript functions that return JSX.
   * Use **hooks (like useState, useEffect)** for state and lifecycle methods.
   * Easier to read, test, and optimize.
   * Example:
   * function Greeting(props) {
   * return <h1>Hello, {props.name}!</h1>;
   * }
2. **Class Components** (Older Approach)
   * Uses ES6 classes extending React.Component.
   * Requires render() method to return JSX.
   * Uses **state and lifecycle methods** (e.g., componentDidMount).
   * Example:
   * class Greeting extends React.Component {
   * render() {
   * return <h1>Hello, {this.props.name}!</h1>;
   * }
   * }

**Key Differences**

| **Feature** | **Functional Component** | **Class Component** |
| --- | --- | --- |
| Syntax | Function-based | Class-based |
| State Handling | Uses useState hook | Uses this.state |
| Lifecycle Methods | Uses hooks (useEffect) | Uses methods (componentDidMount) |
| Performance | Faster, better optimization | Slightly slower |
| Code Simplicity | Shorter, cleaner | More complex |

Since React 16.8, **functional components with hooks** have become the standard.

**Question 2: How Do You Pass Data to a Component Using Props?**

**Props (short for "Properties")** allow data to be passed from a parent component to a child component. Props are **read-only** and help make components reusable.

**How to Pass Props?**

function Greeting(props) {

return <h1>Hello, {props.name}!</h1>;

}

function App() {

return <Greeting name="Desai" />;

}

Here, "Desai" is passed as a **prop** to Greeting, and props.name accesses it.

**Passing Multiple Props**

function UserProfile(props) {

return <h2>{props.name} is {props.age} years old.</h2>;

}

<UserProfile name="Desai" age={22} />;

**Default Props**

Greeting.defaultProps = {

name: "Guest",

};

If no name prop is provided, "Guest" will be used.

**Question 3: What is the Role of render() in Class Components?**

The render() method in a class component:

1. **Returns JSX** – Defines what the UI should display.
2. **Re-renders on State/Prop Change** – Updates the UI when state or props change.
3. **Must Be Defined** – Every class component **must** have a render() method.

**Example:**

class Greeting extends React.Component {

render() {

return <h1>Hello, {this.props.name}!</h1>;

}

}

**Key Points**

* render() **must return a single parent element**.
* It **executes every time** the component updates.
* It **should not modify state** inside itself (causes infinite loops).

With the rise of **functional components and hooks**, class components are **less common** in modern React development.