Question 1: What are components in React? Explain the difference between functional components and class components.

**Components in React:**

In React, **components** are the building blocks of a React application. A component is a reusable and self-contained unit that renders part of the user interface (UI). Components can be written as either **functional components** or **class components**.

**Functional Components:**

* **Definition**: Functional components are JavaScript functions that accept **props** (inputs) and return **JSX** (HTML-like syntax) that defines the UI.
* **Characteristics**:
  + **Stateless**: Initially, functional components did not manage state, but with the introduction of **React Hooks** (from React 16.8), functional components can now manage state and handle side effects.
  + **Simpler Syntax**: Functional components are more concise and easier to understand compared to class components.

**Class Components:**

* **Definition**: Class components are ES6 classes that extend React.Component. They have access to lifecycle methods and can hold local state.
* **Characteristics**:
  + **Stateful**: Class components use this.state to manage local state.
  + **Lifecycle Methods**: Class components can use built-in lifecycle methods like componentDidMount(), componentDidUpdate(), and componentWillUnmount() to handle side effects during different stages of the component’s lifecycle.

Question 2: How do you pass data to a component using props?

**Passing Data to a Component Using Props**

In React, **props** (short for "properties") are a way to pass data from a **parent component** to a **child component**. This allows components to be dynamic and reusable, as they can receive different data each time they are used.

**Key Points:**

1. **What Are Props?**
   * Props are **inputs** to a React component, passed down from a parent component.
   * Props are **read-only** within the child component, meaning the child component cannot modify the data passed to it.
   * Props allow data to flow **one-way** from parent to child in the component tree.
2. **How Props Work:**
   * In the parent component, data is passed to the child component through JSX syntax, where the child component is treated as a custom HTML tag, and the data is passed as attributes.
   * The child component then accesses the data using the props object.
3. **Accessing Props in Child Component:**
   * In **functional components**, props are passed as arguments to the function.
   * In **class components**, props are accessed through this.props.
4. **Why Use Props?**
   * Props help to make components **dynamic and reusable**, as the same component can be used with different data in various parts of an application.
   * They allow for **separation of concerns** by keeping the data management and rendering logic in different components.

• Question 3: What is the role of render() in class components?

**The Role of render() in Class Components**

In React, the **render()** method is a crucial part of **class components**. It is responsible for describing what the UI should look like. When you create a class component in React, the render() method must return a JSX (JavaScript XML) element, which React will use to update the DOM and display the UI.

**Key Points about render():**

1. **Required Method in Class Components**:
   * The render() method is a required method in **class components**. It is automatically called whenever the component's state or props change, triggering a re-render of the component.
   * Unlike functional components, which simply return JSX directly, class components require the render() method to return JSX.
2. **Purpose**:
   * The primary purpose of the render() method is to return the JSX that defines the structure and content of the component's UI.
   * It can also contain logic, such as conditional rendering, to decide what gets displayed based on the component's state or props.
3. **Re-rendering**:
   * Whenever a component's state or props change, React automatically calls the render() method to update the UI.
   * This makes render() essential for ensuring that the UI reflects the most up-to-date data.
4. **Syntax**:
   * The render() method is part of the **React.Component** class, so it can be used inside any class component.
   * It should return a single JSX element or null (if nothing is to be rendered).