**Question 1: What is JSX in React.js? Why is it used?**

**JSX (JavaScript XML)** is a syntax extension for JavaScript used in **React.js** to describe what the UI should look like. It allows developers to write HTML-like code inside JavaScript.

**Why JSX is Used?**

1. **Simplifies UI Development** – Looks like HTML but allows JavaScript functionality.
2. **Better Readability** – Easier to understand compared to React.createElement().
3. **Enhances Performance** – Gets compiled into optimized JavaScript code.
4. **Helps with Debugging** – Provides clear error messages in development.
5. **Supports JavaScript Expressions** – Allows dynamic content rendering.

**Question 2: How is JSX Different from Regular JavaScript? Can You Write JavaScript Inside JSX?**

**Differences from Regular JavaScript:**

1. **JSX Looks Like HTML** – Regular JavaScript requires document.createElement(), but JSX allows writing elements directly.
2. **Attributes Use CamelCase** – Example: <div className="container"> instead of <div class="container">.
3. **Self-Closing Tags Required** – <img src="logo.png" /> instead of <img src="logo.png">.
4. **Curly Braces for JavaScript Expressions** – {variable} inside JSX to render dynamic data.

**Can You Write JavaScript Inside JSX?**

Yes, you can use JavaScript inside JSX using **curly braces {}**.  
Example:

const name = "Desai";

const element = <h1>Hello, {name}!</h1>; // JavaScript inside JSX

However, JSX does not support statements like if-else. Instead, use **ternary operators** or functions.  
Example:

const isLoggedIn = true;

const message = <h2>{isLoggedIn ? "Welcome back!" : "Please log in."}</h2>;

**Question 3: Importance of Using Curly Braces {} in JSX Expressions**

1. **Embed JavaScript Expressions** – {} allows variables, functions, or calculations inside JSX.
2. <h1>{5 + 10}</h1> // Outputs: 15
3. **Dynamic Content Rendering** – Fetch data dynamically.
4. const user = { name: "Desai" };
5. <h2>{user.name}</h2> // Outputs: Desai
6. **Conditional Rendering** – Use ternary operators inside {}.
7. <h3>{isLoggedIn ? "Welcome" : "Guest"}</h3>
8. **Looping Data** – Use .map() for lists.
9. const items = ["Apple", "Banana", "Orange"];
10. <ul>{items.map((item) => <li>{item}</li>)}</ul>
11. **Event Handling** – Call functions dynamically.
12. <button onClick={() => alert("Clicked!")}>Click Me</button>

Curly braces {} make JSX **powerful, flexible, and dynamic**, allowing seamless integration of JavaScript.