1. Define JSX
JSX (JavaScript XML) is a syntax extension for JavaScript used with **React**. It allows developers to write HTML-like code directly inside JavaScript. JSX makes it easier to visualize the UI structure and interact with React components more intuitively.

2. Explain about ECMA Script
ECMAScript (ES) is a standard specification for scripting languages like JavaScript. Managed by **ECMA International**, it defines the rules, syntax, and features for JavaScript.
Key Versions:

- **ES5** (2009): Introduced strict mode, JSON, and array methods.
- **ES6/ES2015**: A major update introducing let, const, arrow functions, classes, template literals, and more.
- Later versions like ES7, ES8, etc., have added features like async/await, Object.entries(), optional chaining, etc.
-

3. Explain React.createElement()
React.createElement() is a core function of React used to create virtual DOM elements. When using JSX, the transpiler (like Babel) converts JSX code into React.createElement().

4. Explain how to create React nodes with JSX
To create a React node using JSX:

1. Use JSX syntax in your React component.
2. Wrap elements inside a component return statement.

JSX can represent:

- DOM elements (<div>, <h1>, etc.)
- React components (<MyComponent />)
- Fragments (<>...</>)

Multiple nodes should be wrapped in a single parent node or a fragment.

5. Define how to render JSX to DOM
JSX elements are rendered to the DOM using ReactDOM.render() (in React 17 and earlier) or through the root API in React 18+.

6. Explain how to use JavaScript expressions in JSX
You can use **JavaScript expressions** inside JSX by wrapping them in curly braces {}.
Example:

```
jsx
const name = 'Salma';
const greeting = <h1>Hello, {name}!</h1>;
```

7. Explain how to use inline CSS in JSX

In JSX, inline styles are defined using a **JavaScript object** and camelCase properties instead of CSS syntax.

Example:

```
jsx
const divStyle = {
  backgroundColor: 'lightblue',
  padding: '20px',
  color: 'white'
};

const element = <div style={divStyle}>Styled with inline CSS!</div>;
```