**10.ReactJS-HOL**

**1. Define JSX**  
JSX (JavaScript XML) is a syntax extension for JavaScript commonly used with React. It allows writing HTML-like code inside JavaScript files. JSX makes it easier to visualize the UI structure in React components. JSX is not required, but it makes code more readable and expressive.

const element = <h1>Hello, world!</h1>;

**2. Explain about ECMA Script**  
ECMAScript (often abbreviated ES) is the standardized scripting language specification upon which JavaScript is based. ES6 (ECMAScript 2015) and later versions introduced new features like classes, modules, arrow functions, let/const, promises, and more.

**3. Explain React.createElement()**  
React.createElement() is a method used to create React elements without using JSX. It takes three arguments: the element type, properties (props), and children.

const element = React.createElement('h1', { className: 'greeting' }, 'Hello, world!');

This creates the same element as:

<h1 className="greeting">Hello, world!</h1>

**4. Explain how to create React nodes with JSX**  
JSX allows creation of React nodes directly by writing HTML-like syntax inside JavaScript.

const heading = <h1>This is a JSX heading</h1>;

const paragraph = <p>This is a paragraph.</p>;

These JSX elements are compiled into React.createElement() calls during the build process.

**5. Define how to render JSX to DOM**  
To render JSX to the DOM, React provides the ReactDOM.render() method.

import React from 'react';

import ReactDOM from 'react-dom';

const element = <h1>Hello, React!</h1>;

ReactDOM.render(element, document.getElementById('root'));

This renders the JSX element into the HTML element with the id root.

**6. Explain how to use JavaScript expressions in JSX**  
JavaScript expressions can be embedded within JSX using curly braces {}.

const name = 'Alice';

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

You can use functions, operations, and other expressions inside JSX:

const element = <p>{10 + 5}</p>; // renders: 15

**7. Explain how to use inline CSS in JSX**  
In JSX, inline styles are provided as an object with camelCase property names:

const divStyle = {

color: 'blue',

backgroundColor: 'lightgray',

padding: '10px'

};

const element = <div style={divStyle}>Styled text</div>;

The style attribute takes a JavaScript object instead of a string.