**1. Define Props**

**Props (short for Properties)** are **read-only inputs** passed from one component (usually a parent) to another (usually a child).

* Used to **pass data and behavior** between components.
* Accessible inside components using props in class components or as a parameter in function components.

**Example:**

function Welcome(props) {

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

}

When used:

<Welcome name="Alice" />

**Key Points:**

* Props are **immutable** (cannot be changed inside the child component).
* Enable **reusability** of components by customizing them with different data.

**2. Explain Default Props**

**Default Props** are used to **set default values** for props in case no value is provided by the parent component.

**Example:**

function Greeting(props) {

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

}

Greeting.defaultProps = {

name: "Guest"

};

If name is not passed, it defaults to "Guest".

**Benefits:**

* Prevents components from breaking if a required prop is missing.
* Useful for setting **fallback values**.

**3. Identify the Differences Between State and Props**

| **Feature** | **Props** | **State** |
| --- | --- | --- |
| Definition | Data passed to a component | Data managed inside the component |
| Mutability | **Immutable** (read-only) | **Mutable** (can change over time) |
| Ownership | Controlled by **parent component** | Controlled by **the component itself** |
| Usage | For configuration or display | For tracking changes (like input, counter) |
| Lifecycle | Set once and remains unchanged unless parent re-renders | Can be updated using setState or useState |

**4. Explain ReactDOM.render()**

**ReactDOM.render()** is the method used to render a React element into the DOM.

**Syntax:**

ReactDOM.render(element, container)

**Parameters:**

* element: The React element or component you want to render.
* container: The DOM node where the React element should be mounted (usually a <div> with an id="root").

**Example:**

import React from 'react';

import ReactDOM from 'react-dom';

ReactDOM.render(<App />, document.getElementById('root'));

**Notes:**

* In modern React (18+), ReactDOM.createRoot() with root.render() is recommended:

import { createRoot } from 'react-dom/client';

const root = createRoot(document.getElementById('root'));

root.render(<App />);