1: JSX and React.createElement

Objective: Understand the difference between JSX and `React.createElement()`.

1. **Using** `React.createElement()`:
   - Create a simple `div` element with an `h1` inside it that says "Hello, React!" using `React.createElement()`.
   - Render this element to the DOM using `ReactDOM.render()`.
2. **Converting to JSX**:
   - Rewrite the same structure using JSX.
   - Compare the readability and ease of writing between JSX and `React.createElement()`.

Bonus:

- Add a `className` of `"greeting"` to the `div` in both versions.

2: Functional and Class Components with Props

Objective: Practice creating components and passing props.

1. **Functional Component**:
   - Create a `UserCard` functional component that accepts `name`, `age`, and `email` as props.
   - Display these props in a styled card (use basic CSS or inline styles).
2. **Class Component**:
   - Convert the `UserCard` into a class component by extending `React.Component`.
   - Ensure it still accepts and displays the same props.
3. **Rendering Components**:
   - In your `App` component, render two instances of `UserCard` with different user data.

Bonus:

- Add a default prop for `email` (e.g., `"Not Provided"`).

3: Handling Events in React

Objective: Practice event handling and passing parameters.

1. **Basic Event Handling**:
   - Create a button that logs "Button clicked!" to the console when clicked.
2. **Passing Parameters**:
   - Modify the button to accept a dynamic parameter (e.g., a user ID) and log it when clicked.
3. **User Input Handling**:
   - Create an input field that updates a state variable (`inputValue`) in real-time as the user types.
   - Display the current value below the input.
4. **Form Submission**:
   - Add a form with a submit button that alerts the entered value when submitted.

Bonus:

- Disable the submit button if `inputValue` is empty.

4: Conditional Rendering and State Management

Objective: Practice using state and conditional rendering in React.

1. **Toggle Visibility**:
   - Create a button that toggles the visibility of a `<p>` element (e.g., "Hello, React!") using `useState`.

- The button text should switch between "Show" and "Hide" based on the state.

2. **Dynamic Styling**:

- Add a checkbox that, when checked, changes the background color of a `<div>` to lightblue.

3. **Conditional Message**:

- If a text input field is empty, display "Please type something".
- Once the user starts typing, show the input value in uppercase.

Bonus:

- Add a "Reset" button that clears the input and resets all states.

5: List Rendering and Component Composition

Objective: Practice rendering lists and composing components.

1. **Dynamic List**:

- Create a state variable `items` initialized as `["Apple", "Banana", "Cherry"]`.
- Render the list as `<ul>` with each item in `<li>`.

2. **Add/Remove Items**:

- Add an input field and a button to append new items to the list.
- Add a "Delete" button next to each item to remove it from the list.

3. **Child Component**:

- Move the `<li>` rendering logic to a separate `ListItem` component.
- Pass each item and a `onDelete` callback as props.

Bonus:

- Prevent adding duplicate items.
- Add a "Clear All" button.