Chapter 4 - Talk is cheap, show me the code

Q.1 Is JSXmandatory for React?
Ans. JSX is not required for using React. Using React without JSX is especially convenient when you don't want to set up compilation in your build environment.

Each JSX element is syntactic sugar for calling React.createElement(component, props, ..children). So, anything you can do with JSX can also be done with just plain JS.

Q.2 Is ES6 mandatory for React?
Ans: 'ES6' is not mandatory for React but it gives us more functionality to write code. The latest react project created on 'ES6', we use 'ES6' in react so, we should be familiar with ES6 features like - arrow functions, classes, spread operator, variables(let and const), 'ES6' stands for ECMAScript6.

If you do not use ES6, you may use the create-react class module instead.

Q.3 {TitleComponent} vs {<TitleComponent />} vs {<TitleComponent><TitleComponent />} in JSX.
Ans: {TitleComponent} : This is a react element or variable or JS expression. We use {} braces to use that in JSX code.
{<TitleComponent />} : This is the way of calling a function component which will return a bunch of elements wrapped under a single parent.
{<TitleComponent><TitleComponent />} : This is same as **{<TitleComponent />}**

Q.4: How can I write comments in JSX?
Ans: Syntax:
{/*   comment   */ }

Q.5 What is <React.Fragment></React.Fragment> or <></>?
Ans: <React.Fragment></React.Fragment> or <></> lets us wrap elements in a single element.

Fragments are useful because grouping elements with a Fragment has no effect on layout or styles, unlike if you wrapped the elements in another container like DOM element.

Q.6 What is Virtual DOM?
Ans: Virtual DOM is the representation of original DOM.

Q.7 What is reconciliation in React?
Ans: Reconciliation is a process where we use a diff algorithm for comparison of two DOM trees. After comparing, creates a new Virtual DOM having the changes. Then it updates the Browser DOM with the least no. of changes possible without rendering into the entire DOM again.

Chapter 4 - Talk is cheap, show me the code

Reconciliation is the process through which React updates the Browser DOM and makes react work faster. React uses diffing algo which has heuristic O(n) time complexity. React would first calculate the difference between the real DOM and virtual DOM, whenever there is an update in the component. React stores a copy of the browser DOM which is called a virtual DOM.
When we make changes or add data, React creates a new virtual DOM and compares it with the previous DOM. Comparison is done by diffing algorithm. React compares the virtual DOM and Real DOM. If it finds any updates, only the changed nodes in Real DOM are updated and the rest remain the same.

Q.8 What is 'React Fiber'?
Ans: Fiber is the new reconciliation engine in React 16. Its main goal is to enable the incremental rendering of Virtual DOM.

Incremental Rendering: the ability to split rendering work intochunks and spread it out over multiple frames.

Other features-

- Ability to pause.
- Abort
- Or re-use work or new updates come in.
- Ability to assign priority to different types of updates.

Q.9 Why do we need keys in React?
Ans: Keys help React identify which items have changed, or added, or removed. Keys should be given to the elements inside the array to give the element a stable identity.

The best way to pick a key is to use IDs from your data as keys:

```
Const todoItems = todos.map((todo) =>
        <li key={index}>
        {todo.text}
        </li>
        );
```

When you don't have stable IDs for rendered items, we may use item index as a key as a last resort. But, React does not recommend using indexes for keys, if the order of items may change.

```
Const todoItems = todos.map((todo, index) =>
        <li key={todo.id}>
        {todo.text}
        </li>
        );
```

Chapter 4 - Talk is cheap, show me the code

Q.10 Can we use index as key?
Ans: When you don't have stable IDs for rendered items, we may use item index as a key as a last resort. But, React does not recommend using indexes for keys, if the order of items may change.

```
Const todoItems = todos.map((todo, index) =>
        <li key={todo.id}>
        {todo.text}
        </li>
        );
```

Q.11 What are 'props' in React?
Ans: props stands for property. Props are arguments passed into React components. They let us flow data from parent component to child component.

Q.12 What is a config-driven UI?
Ans: 'Config-driven UI' are based on configurations of the data application receives. It is rather a good practice to use config driven UIs to make applications dynamic.

It is a very common approach to interact with the user. It provides a generic interface to develop things which help your project scale well. It saves a lot of development time and effort.

Eg: If we have an offer banner on our website and the offer is based on local festivals. Local festivals can be different for different places. In this case, using a config-driven UI is a smart idea to show different offer banners at different places.