Assignment

1. Is JSX mandatory for React?

- NO, If you use this shorthand form for `React.createElement`, it can be almost as convenient to use React without JSX.

```
const e = React.createElement;
const root =
ReactDOM.createRoot(document.getElementById("root"));
root.render(e("div", null, "Hello World"));
```

- Link for check how babel is converted jsx into javascript
- shorturl.at/lmoQ3

2. Is ES6 mandatory for React?

- NO, it is no mandatory

3. How can i write comments in JSX?

```
const Comment = () => {
 return (
   <>
     <h1>Rohit Prajapati</h1>
     {
       // first comment
     }
     {/*
       multi line comment
     */}
     {/* <h2>Hello</h2> */}
   </>
 );
};
```

4. **What is <React.Fragment></React.Fragment> and<></>?**

- key is the only attribute that can be passed to Fragment
- A common pattern in React is for a component to return multiple elements. Fragments let you group a list of children without adding extra nodes to the DOM.

```
const Fragment =()=> {
 return (
   <React.Fragment>
     <ChildA />
     <ChildB />
     <ChildC />
   </React.Fragment>
 );
}
```

- Short Syntax

```
const Fragment =()=> {
 return (
   <>
     <ChildA />
     <ChildB />
     <ChildC />
   </>
 );
}
```

- key is the only attribute that can be passed to Fragment

```
const Fragment =(props)=> {
   return (
     <body>
       {props.list.map(({ title, paragraph }, index) =>
(
         <React.Fragment key={index}>
           <h1>{title} </h1>
           <p>{paragraph}</p>
         </React.Fragment>
       ))}
     </body>
   );
}
```

- The only difference between them is that the shorthand version does not support the key attribute.
- <React.Fragment></React.Fragment> is also a **React Component**
- Empty <></> is also converted to React.Fragement behind the scene

5.What makes DOM manipulation slow?

- The DOM is represented as a tree data structure. Because of that, the changes and updates to the DOM are fast. But after the change, the updated element and it's children have to be re-rendered to update the application UI. The re-rendering or re-painting of the UI is what makes it slow. Therefore, the more UI components you have, the more expensive the DOM updates could be, since they would need to be re-rendered for every DOM update.

6.What is Virtual DOM?

- Defenition1: The virtual DOM (VDOM) is a programming concept where an ideal, or "virtual", representation of a UI is kept in memory and synced with the "real" DOM by a library such as ReactDOM.The Process called Reconciliation.
- Defenition2: The virtual DOM is only a virtual representation of the DOM. Everytime the state of our application changes, the virtual DOM gets updated instead of the real DOM.
- Definition3: ReactJS never updates the original DOM directly(unless a developer use case requires it). In ReactJS, for every DOM object, there will be a corresponding in-memory copy created. This copy is called the Virtual DOM(VDOM).
- In the Virtual DOM tree, each element is represented by a node. A new Virtual DOM tree will be created whenever the element's state changes. The ReactJS's diffing algorithm will compare the current Virtual DOM tree with its previous version. Finally, the VIrtual DOM uses the algorithm to update the actual DOM with the diff algorithm.
- **Link of how virtual dom works animation**:https://res.cloudinary.com/atapas/image/upload/v1649655587/demos/vdom_idrwtz.gif
- Link of the docs: https://blog.greenroots.info/reactjs-virtual-dom-and-reconciliation-explain-like-im-five

7.How is Virtual DOM faster?

- When new elements are added to the UI, a virtual DOM, which is represented as a tree is created. Each element is a node on this tree. If the state of any of these elements changes, a new virtual DOM tree is created. This tree is then compared or "diffed" with the previous virtual DOM tree.
- Once this is done, the virtual DOM calculates the best possible method to make these changes to the real DOM. This ensures that there are minimal operations on the real DOM. Hence, reducing the performance cost of updating the real DOM.

8. What is Reconciliation in React?

- The mechanism to diff one tree with another to determine which parts need to be changed and then update the original DOM with it is called `Reconciliation`.
- Also, diff algo to findout the diffrence between realdom to virtualDOM and update RealDOM(this process called reconciliation)

9.What is React Fiber?

- ReactJS uses a new reconciliation engine called `Fiber` since version 16.0.
- Doc link : https://github.com/acdlite/react-fiber-architecture

10.Why we need keys in react? When do we need keys in React?

- React supports a `key` attribute. When children have keys, React uses the key to match children in the original tree with children in the subsequent tree.
- you can add a new ID property to your model or hash some parts of the content to generate a key. The key only has to be unique among its siblings, not globally unique.
- Keys help React identify which items have changed, are added, or are removed. Keys should be given to the elements inside the array to give the elements a stable identity:

11.Can we use index as a Key in react?

- you can pass an item's index in the array as a key. This can work well if the items are never reordered, but reorders will be slow.
- Reorders can also cause issues with component state when indexes are used as keys. Component instances are updated and reused based on their key. If the key is an index, moving an item changes it. As a result, component state for things like uncontrolled inputs can get mixed up and updated in unexpected ways.
- Link : https://robinpokorny.com/blog/index-as-a-key-is-an-anti-pattern/

12.What is Props in React? ways to

- This function is a valid React component because it accepts a single "props" (which stands for properties) object argument with data and returns a React element. We call such components "function components" because they are literally JavaScript functions.
- Props are Read only you can not modified it
- To pass props, add them to the JSX, just like you would with HTML attributes.
- To read props, use the function Avatar({ person, size }) destructuring syntax.
- You can specify a default value like size = 100, which is used for missing and undefined props.
- You can forward all props with <Avatar {...props} /> JSX spread syntax, but don't overuse it!
- Nested JSX like <Card><Avatar /></Card> will appear as Card component's children prop.

- Props are read-only snapshots in time: every render receives a new version of props.
- You can’t change props. When you need interactivity, you’ll need to set state.

13. What is Config Driven UI?

- Articles link of config Driven UI : https://iamrajatsingh1.medium.com/config-driven-ui-c8e93b730993
- Form on config Driven ui link: https://data-driven-forms.org/