Set 4 : JSX and HTML

Section A: Basic JSX + HTML Structure (Q1–Q15)

1. Create a `Greeting` component that returns a `<h1>` with the text "Hello, React Beginner!"
2. Write a component `MyIntro` that returns your name and city inside a `<div>` with two `<p>` tags.
3. Create a `HobbyList` component that shows a list of 3 hobbies using `<ul>` and `<li>`.
4. Display a favorite quote inside a `<blockquote>` in a component named `QuoteBox`.
5. Return 5 different HTML tags (`h1`, `p`, `span`, `b`, `u`) in a single component.
6. Make a component that renders an `<img>` with your favorite meme using its URL.
7. Create a `FoodItem` component that returns "I love Biryani" using JSX and a `<strong>` tag.
8. Use `<br />` and `<hr />` to format content inside a component named `InfoCard`.
9. Write a `SocialLinks` component that returns 3 anchor `<a>` tags for GitHub, LinkedIn, and Twitter.
10. Make a component that shows today's date using `{new Date().toDateString()}`.
11. Create a `CourseCard` component with `Course Name`, `Instructor`, and `Duration` in headings.
12. Display a math expression result (e.g. `2 + 2 = 4`) using `{}` in JSX.

13. Return a complete address using multiple `<span>` tags in a `MyAddress` component.
14. Render a simple navigation bar using JSX (without functionality), just links using `<nav>`.
15. Use `React.Fragment` to return 3 sibling headings in a `PageTitles` component.

🔁 Section B: Components with Props (Q16–Q35)

1. Create a `UserCard` component that accepts `name` and `email` as props and displays them.
2. Build a `CarDetails` component that takes `brand`, `model`, and `year` props.
3. Create a component `SimpleMath` that takes two number props and shows their sum.
4. Make a `WelcomeMessage` component that accepts `username` and displays "Hello, [username]!"
5. Pass a quote and author to a `QuoteCard` component via props and show both.
6. Create a `PetProfile` component with props: `petName`, `type`, and `age`.
7. Build a `Book` component that takes `title`, `author`, and `pages` as props.
8. Render multiple `Book` components from a parent `Library` component with different props.
9. Create a `SkillTag` component that takes one prop `skill` and returns it inside a `<button>`.
10. Make a `ProfilePic` component that takes an `imgUrl` prop and renders an image.

> ✅ Important: Props are read-only. You cannot change them inside the child component.

1. Create a `GreetingTime` component that takes a `time` prop and returns "Good Morning" if time < 12.
2. Pass an array of subjects to a `SubjectsList` component and render them using `.map()`.
3. Pass an object prop (e.g. `user={{name: "Aman", age: 22}}`) and access the object fields in JSX.
4. Create a component that receives 3 strings as props and joins them in one paragraph.
5. Pass a boolean prop to a component and show conditional JSX: "Active" or "Inactive".
6. Make a component that returns "You scored [marks] marks!" using props.
7. Create a component `MoviePoster` that shows an image and title using props.
8. Pass JSX as a prop using `children` and render it in a `CardWrapper` component.
9. Make a `UserBadge` component with `name`, `email`, and `status` props.
10. Create a `WeatherBox` component that takes `city` and `temperature` and displays them together.

> 💡 Suggestion: Use object destructuring for cleaner props access:

```
function WeatherBox({ city, temperature }) {
  return <p>{city} is {temperature}°C today.</p>;
}
```

🔗 Section C: Nested Components & Conditional JSX (Q36–Q50)

1. Create a `Dashboard` component that renders 3 child components: `Profile`, `Stats`, and `LogoutButton`.
2. Render a `StatusMessage` component that accepts a boolean `isOnline` and returns "Online" or "Offline".
3. Build a `Result` component that accepts a `score` prop and shows "Pass" if score > 40 else "Fail".
4. Write a `BirthdayMessage` component that checks if `isTodayBirthday` is true and returns a special wish.
5. Create a `LaptopDetails` component with props like `brand`, `price`, `ram`.
6. Use a ternary operator inside JSX to show different emojis based on `mood` prop.
7. Create a `FlagIcon` component that takes a `countryCode` prop and shows flag emoji.
8. Make a `LanguageGreeting` that returns different greetings based on language prop: 'hi', 'en', 'fr'.
9. Render a `ListItem` component inside a loop from parent and pass `id`, `text` as props.
10. Build a `ProductList` with a list of products passed as props. Render their names using `.map()`.

> ✅ Important: In JSX, always use `className` instead of `class` for adding classes.

1. Create a `PetList` component that takes an array of pet objects and renders them.
2. Write a `ReviewCard` that takes reviewer name, comment, and stars.
3. Render a `PriceTag` that takes `price` and `currency` as props and returns `₹100` or `$100`.

4. Write a `ShowName` component that returns "Hello Guest" if no name prop is passed.

5. Use default props in a component so it renders a default value when no prop is given.

🧠 BONUS: MCQs on JSX + Props (5 Questions)

Q1. What does the following JSX return?

```
function App() {
  return <h1>Hello World</h1>;
}
```

- A. JavaScript
- B. HTML
- C. JSX
- D. Component ✅ Answer: C

Q2. Which attribute is used instead of `class` in JSX? - A. id - B. classname - C. className - D. classes ✅ Answer: C

Q3. Props in React are: - A. Mutable - B. Read-only - C. Can only be used in parent - D. Replaced by variables ✅ Answer: B

Q4. Which of the following is the correct way to pass props?

```
<User name="John" />
```

- A. `<User name: "John" />`
- B. `<User name="John" />`
- C. `<User name='John'>`
- D. `User name="John"` ✅ Answer: B

Q5. Can a component return multiple JSX elements without a wrapper? - A. No - B. Only with `div` - C. Only with `section` - D. Yes, using React Fragments ✅ Answer: D