---

1. What is the Virtual DOM?

The **Virtual DOM (VDOM)** is a lightweight, in-memory representation of the real DOM used by React. It is a JavaScript object that mirrors the structure of the actual DOM.

Instead of directly updating the real DOM whenever data changes, React first updates the Virtual DOM. The Virtual DOM is much faster to manipulate because it exists only in memory and does not trigger browser reflows or repaints.

Whenever a change occurs:

1. React creates a new Virtual DOM tree
2. It compares the new tree with the previous one (called **diffing**)
3. It calculates the minimum number of changes required
4. Only those changes are applied to the real DOM

This approach significantly improves performance and makes UI rendering efficient.

---

2. What is the difference between the ES6 and ES5 standards?

ES5 and **ES6 (ES2015)** are versions of the ECMAScript standard used to write JavaScript.

ES5:

- Introduced in 2009
- Uses var for variable declaration
- No classes
- No arrow functions
- Callback-based programming

Example:

```
var x = 10;

function add(a, b) {

  return a + b;

}
```

ES6:

- Introduced in 2015
- Introduces let and const
- Supports classes
- Arrow functions
- Modules

- Promises

Example:

```
const add = (a, b) => a + b;
```

Key Differences:

ES5	**ES6**
var only	let, const
No classes	Class syntax
No modules	Import/export
Verbose code	Cleaner syntax

ES6 is preferred in React because it makes code cleaner and more readable.

---

3. What is an arrow function and how is it used in React?

An **arrow function** is a shorter syntax for writing functions introduced in ES6.

Syntax:

```
const functionName = () => {
  // code
};
```

Example:

```
const greet = () => {
  return "Hello";
};
```

Usage in React:

Arrow functions are widely used in React for:

- Event handling
- Functional components
- Avoiding this binding issues

Example in React:

```
const Button = () => {
  return <button>Click Me</button>;
};
```

Arrow functions automatically bind this, which simplifies React code and avoids errors.

---

4. What are the components in React?

Components are the building blocks of a React application. A component represents a part of the user interface.

Each component:

- Is reusable
- Has its own logic and UI
- Can receive data via props

Types of Components:

1. **Class Components**
2. **Functional Components**

Example:

```
function Welcome() {
  return <h1>Hello React</h1>;
}
```

Components help in breaking the UI into smaller, manageable pieces.

---

5. What are the differences between class and functional components?

Class Components:

- Use ES6 classes
- Use this
- Support lifecycle methods
- Use state

Example:

```
class Hello extends React.Component {
  render() {
    return <h1>Hello</h1>;
  }
}
```

Functional Components:

- Simple JavaScript functions
- No this
- Use Hooks for state and lifecycle

- More readable

Example:

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

Differences:

Class	**Functional**
More code	Less code
Uses this	No this
Lifecycle methods	Hooks
Slower	Faster

Functional components are preferred in modern React.

---

6. What are the advantages of using React?

React provides several advantages for modern web development:

1. **Reusable components**
2. **Fast rendering using Virtual DOM**
3. **One-way data binding**
4. **Easy to maintain**
5. **Strong community support**
6. **SEO friendly**
7. **Used for single-page applications**

React makes UI development efficient, scalable, and maintainable.

---

7. What is the Virtual DOM? How does React use it to render the UI?

The **Virtual DOM** is a copy of the real DOM stored in memory.

How React uses it:

1. Initial UI is rendered using Virtual DOM
2. On state change, React creates a new Virtual DOM
3. React compares old and new Virtual DOM (diffing)
4. React updates only changed parts in real DOM

Benefits:

- Faster updates
- Better performance
- Efficient rendering

This process is called **reconciliation**.

---

8. What are the lifecycle methods going to be deprecated in React v16?

React v16 deprecated some lifecycle methods because they were unsafe for async rendering.

Deprecated methods:

- componentWillMount()
- componentWillReceiveProps()
- componentWillUpdate()

Reason:

- Caused bugs in async rendering
- Poor performance
- Hard to maintain

Replacement:

- componentDidMount()
- getDerivedStateFromProps()
- componentDidUpdate()

These changes improved stability and performance.

---

9. What are modules? Explain its types.

Modules allow JavaScript code to be split into separate files and reused.

Types of Modules:

ES6 Modules:

```
export default function add(a, b) {
  return a + b;
}
import add from './math.js';
```

CommonJS Modules:

```
module.exports = add;
```

Modules improve:

- Code organization
- Reusability
- Maintainability

React applications heavily use ES6 modules.

---

10. What objects are in React? Explain.

React uses several important objects:

1. React Object

- Core React library
- Used to create components

2. Component Object

- Base class for class components

3. Props Object

- Used to pass data to components
- Read-only

4. State Object

- Stores component data
- Mutable

5. Refs Object

- Used to access DOM elements directly

Example:

this.state = { count: 0 };

These objects help manage data, UI, and behavior in React applications.