

Question 1

Not yet answered

Marked out of 1.00

What will happen when you call a Hook like `useEffect` inside a conditional block in a React functional component?

```
function MyComponent({ flag }) {  
  if (flag) {  
    useEffect(() => {  
      console.log("Effect ran");  
    }, []);  
  }  
  return <div>Hello</div>;  
}
```

- ☐ a. React will log a warning but proceed without errors.
- ☐ b. React will throw an error because Hooks must be called unconditionally.
- ☐ c. The effect will run only when flag is true.
- ☐ d. The effect will be skipped silently when flag is false.

Question 2

Not yet answered

Marked out of 1.00

```
const Child = React.memo(({ obj }) => {  
  console.log("Rendered");  
  return <div>{obj.count}</div>;  
});  
function App() {  
  function App() {  
    const [count, setCount] = React.useState(0);  
    const obj = { count };  
    return (  
      return (  
        <>  
          <Child obj={obj} />  
          <button onClick={() => setCount(count + 1)}>Increment</button>  
        </>  
      );  
    );  
  }  
}
```

What is printed to the console each time the button is clicked?

- ☐ a. A new object reference is created on each render, causing re-render.
- ☐ b. `React.memo` uses deep comparison, and deep objects always differ.
- ☐ c. `React.memo` triggers re-render due to `console.log` side-effect.
- ☐ d. `React.memo` doesn't support object props.

Question 3

Not yet answered

Marked out of 1.00

```
function ErrorFallback() {
  return <div>Error occurred</div>;
}
function Component() {
function Component() {
  throw new Error("Something went wrong");
}
function App() {
function App() {
  return (
    <React.Suspense fallback={<div>Loading...</div>}>
      <Component />
    </React.Suspense>
  );
}
}
```

What does React.Suspense catch and handle internally?

- ☐ a. Runtime JavaScript errors in <Component />
- ☐ b. Promises thrown during rendering, such as from React.lazy
- ☐ c. Errors in useEffect or asynchronous handlers
- ☐ d. Failed fetch requests by default

Question 4

Not yet answered

Marked out of 1.00

What is logged when the button is clicked the first time?

```
function App() {
  const [a, setA] = React.useState(0);
  const [b, setB] = React.useState(0);

  function handleClick() {
    setA(a + 1);
    setB(b + 1);
    console.log(a, b);
  }

  return <button onClick={handleClick}>Click</button>;
}
```

- ☐ a. React throws an error
- ☐ b. The updated values of a and b
- ☐ c. 0 0
- ☐ d. 1 1

Question 5

Not yet answered

Marked out of 1.00

Which value will be printed to the console when the following component's button is clicked once?

```
function App() {  
  const [count, setCount] = React.useState(0);  
  
  function handleClick() {  
    setTimeout(() => {  
      console.log("Count is:", count);  
    }, 1000);  
    setCount(count + 1);  
  }  
  
  return <button onClick={handleClick}>Click</button>;  
}
```

- ☐ a. Count is: undefined
- ☐ b. Count is: 1
- ☐ c. Count is: NaN
- ☐ d. Count is: 0