
Started on Monday, 26 May 2025, 12:27 PM

State Finished

Completed on Monday, 26 May 2025, 12:46 PM

Time taken 19 mins 12 secs

Marks 4.00/5.00

Grade 80.00 out of 100.00

Question 1

Complete

Mark 1.00 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 throw an error because Hooks must be called unconditionally.
- ☐ b. React will log a warning but proceed without errors.
- ☐ c. The effect will run only when flag is true.
- ☐ d. The effect will be skipped silently when flag is false.

Question 2

Complete

Mark 1.00 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. React.memo uses deep comparison, and deep objects always differ.
- ☒ b. A new object reference is created on each render, causing re-render.
- ☐ c. React.memo doesn't support object props.
- ☐ d. React.memo triggers re-render due to console.log side-effect.

Question 3

Not 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. Failed fetch requests by default
- ☐ b. Runtime JavaScript errors in <Component />
- ☐ c. Promises thrown during rendering, such as from React.lazy
- ☐ d. Errors in useEffect or asynchronous handlers

Question 4

Complete

Mark 1.00 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. 0 0
- ☐ c. 1 1
- ☐ d. The updated values of a and b

Question 5

Complete

Mark 1.00 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: NaN
- ☐ b. Count is: 1
- ☐ c. Count is: undefined
- ☒ d. Count is: 0