

Time left 0:14:45

**Question 1**

Not yet answered

Marked out of 1.00

What is logged repeatedly in this example?

```
function useLogger(value) {  
  React.useEffect(() => {  
    const id = setInterval(() => {  
      console.log("Value is:", value);  
    }, 1000);  
    return () => clearInterval(id);  
  }, []);  
}
```

```
function App() {  
  const [count, setCount] = React.useState(0);  
  useLogger(count);  
  return <button onClick={() => setCount(count + 1)}></button>;  
}
```

- ☐ a. A runtime error
- ☐ b. Undefined
- ☐ c. Always 0
- ☐ d. Updated count value each second

**Question 2**

Not yet answered

Marked out of 1.00

What is printed every second after clicking the button a few times?

```
function App() {  
  const [count, setCount] = React.useState(0);  
  const log = () => {  
    console.log("Count is:", count);  
  };  
  React.useEffect(() => {  
    const id = setInterval(log, 1000);  
    return () => clearInterval(id);  
  }, []);  
}
```

```
  return <button onClick={() => setCount(count + 1)}></button>;  
}
```

- ☐ a. Always 0
- ☐ b. Increments after every click
- ☐ c. The latest count value
- ☐ d. A different number each second

**Question 3**

Not yet answered

Marked out of 1.00

What is printed to the console on the first button click?

```
function App() {  
  const [state, setState] = React.useState(0);  
  const ref = React.useRef(0);  
  const handleClick = () => {  
    ref.current += 1;  
    setState(state + 1);  
    console.log("State:", state, "Ref:", ref.current);  
  };  
}
```

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

- ☐ a. State: 1 Ref: 2
- ☐ b. State: 1 Ref: 1
- ☐ c. State: 0 Ref: 1
- ☐ d. State: 1 Ref: 0

**Question 4**

Not yet answered

Marked out of 1.00

What will be logged to the console when the following component is rendered inside `<React.StrictMode>`?

```
function App() {  
  React.useEffect(() => {  
    console.log("Effect ran");  
  }, []);  
}
```

```
  return <div>Hello</div>;  
}
```

- ☐ a. Nothing
- ☐ b. Effect ran
- ☐ c. Effect ran (logged twice)
- ☐ d. Compilation error

**Question 5**

Not yet answered

Marked out of 1.00

Why does the following component cause an infinite render loop?

```
function App() {  
  const [count, setCount] = React.useState(0);  
  const obj = {  
    increment: () => setCount(count + 1),  
  };  
  React.useEffect(() => {  
    obj.increment();  
  }, [obj]);  
}
```

```
  return <div>{count}</div>;  
}
```

- ☐ a. Because increment modifies state incorrectly
- ☐ b. Because obj is re-created on every render
- ☐ c. There is no loop
- ☐ d. Because count changes inside useEffect