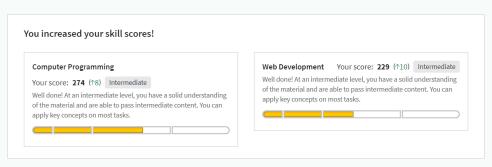
Congratulations! You passed!

Grade Latest Submission received 100% Grade 100%

To pass 80% or higher

Go to next item



1. You are building a form using both Formik and Yup libraries, where one of the inputs is an email. Here are this input's client validation rules:

1 / 1 point

- It has to be a valid email address.
- If the email input is invalid, a message "Invalid email address" will be displayed.
- If the email input is blank, a message "Required" will be shown.

Based on the above requirements, choose the correct Yup validation code from the provided options.



1 Yup.email().string("Invalid email address").required("Required")

1 Yup.email("Invalid email address").required("Required").

Correct, first Yup needs to know the type of value (string) and then chain the different validation rules with their associated error message to show.

2. You have the following React application where you have a ToDo component that has two text labels and an uncontrolled text input and the entry point App component that renders a list of two ToDos and a button to reverse the order of the ToDos. To avoid a React keys warning, a key is provided to each ToDo component, with the index as its value. Suppose that the next sequence of events happen in the application:

1/1 point

- 1. You write "Wash dishes" in the first ToDo input
- 2. You write "Buy groceries" in the second ToDo input
- 3. You click the button to reverse the order

What would happen on the screen after that?

```
14
15
16
      function App() {
17
        const [todos, setTodos] = useState([
18
            id: 'todo1',
createdAt: '18:00',
19
20
21
22
23
            id: 'todo2',
createdAt: '20:30',
24
25
26
       ]);
27
28
       const reverseOrder = () => {
29
          // Reverse is a mutative operation, so we need to create a new array first.
30
31
          setTodos([...todos].reverse());
32
33
34
       return (
          <div>
35
            <button onClick={reverseOrder}>Reverse</button>
36
37
            {todos.map((todo, index) => (
              <ToDo key={index} id={todo.id} createdAt={todo.createdAt} />
38
            ))}
39
       );
40
```

O todo2 Buy groceries 20:30

todo1 Wash dishes 18:00

• todo2 Wash dishes 20:30 todo1 Buy groceries 18:00

O todo1 Buy groceries 18:00

todo2 Wash dishes 20:30

⊘ Correct

Correct, when reversing the order React understands they are still the same nodes with key=1 and key=2, so it will preserve their internal state (input value). Since the props are different though, it will just update the node with the new prop values.

3. A team is tasked with implementing a ThemeProvider for an application that will inject into the tree a light/dark theme, as well as a toggler function to be able to change it. For that, the following solution code has been incorporated. However, unwittingly, some errors have been introduced that prevent it from working correctly. What are the errors in the code solution? Select all that apply.

1/1 point

1/1 point

```
17 };
```

The children are not passed through

Correct
Correct, ThemeProvider should use the children prop and pass it as a direct child of ThemeContext. Provider.

The default value of createContext should be "light" instead of undefined.

☐ There is no need to use local state for the context.

✓ The toggleTheme implementation is incorrect.

✓ Correct
Correct, it should be instead toggleTheme: () =>setTheme(theme === "light" ? "dark" : "light").



17 const MyButton = withClick(Button); 18 export default function App() { 20 return <MyButton onClick={() => console.log("AppClick")}>Submit</MyButton>; 21

O True

False

⊘ Correct

Correct. It is false to claim that the WithClick string will never be output to the console. Actually, due to the order of the spread operator in the different components, the withclick Higher-order component (HOC) takes precedence, and is the thing to be console logged.

When writing a test for a React component using jest and react-testing-library, how would you assert that a function has been called with some specific arguments?

- O Using the toHaveAttribute matcher.
- Using the toHaveBeenCalledWith matcher.
- O Using the toHaveBeenCalled matcher.
- **⊘** Correct

Correct, this is the proper matcher to check the arguments of the function call.

7. Is the following piece of code a valid implementation of the render props pattern?

```
<MealProvider render={data => (
 Ingredients: {data.ingredients}
)}/>
```

Yes

O No

⊘ Correct

Correct, it uses a render type prop that is a function that returns JSX.

8. Inspect the given code snippet.

React.useEffect(()=> {
 console.log('The initial value of the price variable is', price)

Where should you add an empty array to have the effect ran only on initial render?

- As a second argument of the arrow function passed to the useEffect() call.
- O Vou need to add an empty array in a senarate arrow function

1/1 point

1/1 point

1/1 point

1/1 point

O You can't add an empty array in this code snippet.

Correct

Correct. You need to add it as a second argument of the arrow function passed to the useEffect() call.

9. You are given the below piece of code.

```
import {useState} from "react";
      export default function App() {
   const [restaurantName, setRestaurantName] = useState("Lemon");
         function updateRestaurantName() {
   setRestaurantName("Little Lemon");
 8
9
         };
10
11
12
         return (
            <div>
              <h1>{restaurantName}</h1>
13
               <button onClick={updateRestaurantName}>
14
15
              Update restaurant name </button>
            </div>
17
18
        );
      };
```

 $True\ or\ false: The\ \textbf{restaurantName}\ variable's\ value\ will\ always\ be\ reset\ between\ re-renders\ of\ the\ App\ component.$

O True

False

⊘ Correct

 ${\tt Correct. The\ restaurant Name\ variable's\ value\ will\ not\ be\ reset\ between\ re-renders\ of\ the\ App\ component.}$

10. Is this valid code?

```
if (data !== '') {
    useEffect(() => {
        | setData('test data');
        | });
}
```

No

O Yes

⊘ Correct

Correct. If you use a hook in a condition, you're breaking rules! Thus, the below code is invalid.

1/1 point

1/1 point