
Started on Wednesday, 8 September 2021, 4:11 PM

State Finished

Completed on Wednesday, 8 September 2021, 4:13 PM

Time taken 1 min 23 secs

Grade 3.00 out of 7.00 (43%)

Question 1

Incorrect

Mark 0.00 out of 1.00

A functional (or function) component can only be created using the **function** keyword.

Select one:

☒ True ✖

☐ False

Incorrect. Functional components may also be created using the arrow function syntax, which does not use the function keyword.

The correct answer is 'False'.

Question 2


Correct

Mark 1.00 out of 1.00

When the unary plus operator `+` is placed in front of a string, and that string is only operand, for example, like this:

It will:

Select one:

- ☐ a. Double the string.
- ☒ b. Turn the string into a number, if it can be converted to a number, or convert it to the special value *NaN* if it cannot be converted to a number. 
- ☐ c. Turn the string into a number, if it can be converted to a number, or convert it to Boolean *false* if it cannot be converted to a number.
- ☐ d. Add one to the number inside it.

Your answer is correct.

The correct answer is: Turn the string into a number, if it can be converted to a number, or convert it to the special value *NaN* if it cannot be converted to a number.

Question 3

Partially correct

Mark 2.00 out of 3.00

Which of these following examples would work as a return from a functional component?

Select one or more:

☐ a.

```
function Example(props) {  
  return (  
    <p>{this.props.name}</p>  
  );  
}
```

☐ b.

```
function Example() {  
  return <div />;  
}
```

☒ c.

```
function Example() {  
  return (  
    <div>Hello</div>  
    <div>World</div>  
  );  
}
```

✗ Incorrect. You cannot return two top-level JSX elements from a component.

☒ d.

```
function Example() {  
  return (  
    <div>  
      <p>Hello World</p>  
    </div>  
  );  
}
```

✓ Correct. You can return multiple JSX elements as long as there's only one JSX element at the top level.

☒ e.

```
const Example = ({name}) => {  
  return (  
    <p>{name}</p>  
  );  
};
```



Correct. Assuming that **name** is a valid property of **props**, the object destructuring syntax makes the variable **name** available to be used within this component.

☐ f.

```
const Example = () => {  
  return (  
    <span>{props.name}</span>  
  );  
};
```

Your answer is partially correct.

You have correctly selected 2.

The correct answers are:

```
function Example() {  
  return <div />;  
}
```

```
function Example() {  
  return (  
    <div>  
      <p>Hello World</p>  
    </div>  
  );  
}
```

```
const Example = ({name}) => {  
  return (  
    <p>{name}</p>  
  );  
};
```

Question 4

Incorrect

Mark 0.00 out of 1.00

When do you use the **render** attribute instead of **component** in a react-router <Route> component?

Select one:

- ☐ a. When you need to pass the react-router match.params object.
- ☐ b. When you need to pass props to the component being routed to.
- ☒ c. When you are routing to a class component only.
- ☐ d. Whenever you are using an exact path for the <Route>.

✗ Incorrect. You may be thinking that class components use the render() method, and that is true, but that is unrelated to this question.

Your answer is incorrect.

The correct answer is: When you need to pass props to the component being routed to.

Question 5

Incorrect

Mark 0.00 out of 1.00

Presentational components do not ever contain any local state information.

Select one:

☒ True ✖

☐ False

Incorrect. Presentational containers can contain local state information related to the UI, such as if a modal is hidden or visible.

The correct answer is 'False'.



