React.js

Q1. If you want to import just the Component from the React library, what syntax do you use?

- ☐ `import React.Component from 'react'`
- ☐ `import [ Component ] from 'react'`
- ☐ `import Component from 'react'`
- ☑ `import { Component } from 'react'`

Q2. If a function component should always render the same way given the same props, what is a simple performance optimization available for it?

- ☑ Wrap it in the `React.memo` higher-order component.
- ☐ Implement the `useReducer` Hook.
- ☐ Implement the `useMemo` Hook.
- ☐ Implement the `shouldComponentUpdate` lifecycle method.

Q3. How do you fix the syntax error that results from running this code?

```
const person =(firstName, lastName) =>
{
  first: firstName,
  last: lastName
}
console.log(person("Jill", "Wilson"))
```

- ☑ Wrap the object in parentheses.
- ☐ Call the function from another file.
- ☐ Add a return statement before the first curly brace.
- ☐ Replace the object with an array.

Q4. If you see the following import in a file, what is being used for state management in the component?

```
 import React, {useState} from 'react';
```

- ☑ React Hooks
- ☐ stateful components
- ☐ math
- ☐ class components

Q5. Using object literal enhancement, you can put values back into an object. When you log person to the console, what is the output?

```
const name = 'Rachel';
const age = 31;
const person = { name, age };
console.log(person);
```

- ☐ `{{name: "Rachel", age: 31}}`
- ☑ `{name: "Rachel", age: 31}`
- ☐ `{person: "Rachel", person: 31}}`
- ☐ `{person: {name: "Rachel", age: 31}}`

Q6. What is the testing library most often associated with React?

- ☐ Mocha
- ☐ Chai
- ☐ Sinon
- ☑ Jest

Reference

Q7. To get the first item from the array ("cooking") using array destructuring, how do you adjust this line?

```
const topics = ['cooking', 'art', 'history'];
```

- ☐ `const first = ["cooking", "art", "history"]`
- ☐ `const [] = ["cooking", "art", "history"]`
- ☐ `const [, first]["cooking", "art", "history"]`
- ☑ `const [first] = ["cooking", "art", "history"]`

Q8. How do you handle passing through the component tree without having to pass props down manually at every level?

- ☐ React Send
- ☐ React Pinpoint
- ☐ React Router
- ☑ React Context

Reference

Q9. What should the console read when the following code is run?

```
const [, , animal] = ['Horse', 'Mouse', 'Cat'];
console.log(animal);
```

- ☐ Horse
- ☑ Cat
- ☐ Mouse
- ☐ undefined

Q10. What is the name of the tool used to take JSX and turn it into createElement calls?

- ☐ JSX Editor
- ☐ ReactDOM
- ☐ Browser Buddy
- ☑ Babel

Q11. Why might you use useReducer over useState in a React component?

- ☐ when you want to replace Redux
- ☑ when you need to manage more complex state in an app
- ☐ when you want to improve performance
- ☐ when you want to break your production app

Q12. Which props from the props object is available to the component with the following syntax?

```
<Message {...props} />
```

- ☐ any that have not changed
- ☑ all of them
- ☐ child props
- ☐ any that have changed

Q13. Consider the following code from React Router. What do you call :id in the path prop?

```
<Route path="/:id" />
```

- ☐ This is a route modal
- ☑ This is a route parameter
- ☐ This is a route splitter
- ☐ This is a route link

Q14. If you created a component called Dish and rendered it to the DOM, what type of element would be rendered?

```
function Dish() {
  return <h1>Mac and Cheese</h1>;
}

ReactDOM.render(<Dish />, document.getElementById('root'));
```

- ☐ `div`
- ☐ section
- ☐ component
- ☑ `h1`

Q15. What does this React element look like given the following code? (Alternative: Given the following code, what does this React element look like?)

```
React.createElement('h1', null, "What's happening?");
```

- ☐ `<h1 props={null}>What's happening?</h1>`
- ☑ `<h1>What's happening?</h1>`
- ☐ `<h1 id="component">What's happening?</h1>`
- ☐ `<h1 id="element">What's happening?</h1>`

Reference

Q16. What property do you need to add to the Suspense component in order to display a spinner or loading state?

```
function MyComponent() {
  return (
    <Suspense>
      <div>
        <Message />
      </div>
    </Suspense>
  );
}
```

- ☐ lazy
- ☐ loading
- ☑ fallback
- ☐ spinner

Reference

Q17. How would you describe the message variable wrapped in curly braces below?

```
const message = 'Hi there';
const element = <p>{message}</p>;
```

- ☐ a JS function
- ☐ a JS element
- ☑ a JS expression
- ☐ a JSX wrapper

Q18. What can you use to handle code splitting?

- ☐ `React.memo`
- ☐ `React.split`
- ☑ `React.lazy`
- ☐ `React.fallback`

Reference

Q19. When do you use `useLayoutEffect`?

- ☐ to optimize for all devices
- ☐ to complete the update
- ☑ to change the layout of the screen
- ☐ when you need the browser to paint before the effect runs

[Source] (https://react.dev/reference/react/useLayoutEffect) "useLayoutEffect is a version of useEffect that fires before the browser repaints the screen."

[Explanation]The correct answer to the question "When do you use useLayoutEffect?" is: When you need to change the layout of the screen.

useLayoutEffect is used when you need to perform DOM mutations that rely on the updated layout of the elements. It allows you to make changes to the DOM synchronously before the browser performs its painting step. This can be useful when you need to measure or manipulate the layout, such as accessing element dimensions or positions, calculating scroll offsets, or performing other operations that require up-to-date layout information.

The other option provided as answer is not accurate:

"When you need the browser to paint before the effect runs" is not correct. The purpose of useLayoutEffect is to run the effect synchronously after the DOM updates but before the browser paints, allowing you to make layout-related changes before the visual rendering occurs.

Explanation: `useLayoutEffect` gets executed *before* the `useEffect` hook without much concern for DOM mutation. Even though the React hook `useLayoutEffect` is set after the `useEffect` Hook, it gets triggered first!

Q20. What is the difference between the click behaviors of these two buttons (assuming that this.handleClick is bound correctly)?

```
A. <button onClick={this.handleClick}>Click Me</button>
B. <button onClick={event => this.handleClick(event)}>Click Me</button>
```

- ☐ Button A will not have access to the event object on click of the button.
- ☐ Button B will not fire the handler this.handleClick successfully.
- ☐ Button A will not fire the handler this.handleClick successfully.
- ☑ There is no difference.

Q21. How do you destructure the properties that are sent to the Dish component?

```
function Dish(props) {
  return (
    <h1>
      {props.name} {props.cookingTime}
    </h1>
  );
}
```

- ☐ `function Dish([name, cookingTime]) { return <h1>{name} {cookingTime}</h1>; }`
- ☑ `function Dish({name, cookingTime}) { return <h1>{name} {cookingTime}</h1>; }`
- ☐ `function Dish(props) { return <h1>{name} {cookingTime}</h1>; }`
- ☐ `function Dish(...props) { return <h1>{name} {cookingTime}</h1>; }`

Q22. When might you use `React.PureComponent`?

- ☐ when you do not want your component to have props
- ☐ when you have sibling components that need to be compared
- ☑ when you want a default implementation of `shouldComponentUpdate()`
- ☐ when you do not want your component to have state

Reference

Q23. Why is it important to avoid copying the values of props into a component's state where possible?

- ☐ because you should never mutate state
- ☐ because `getDerivedStateFromProps()` is an unsafe method to use
- ☑ because you want to allow a component to update in response to changes in the props
- ☐ because you want to allow data to flow back up to the parent

Q24. What is the children prop?

- ☐ a property that adds child components to state
- ☑ a special property that JSX creates on components that contain both an opening tag and a closing tag, referencing it's contents.
- ☐ a property that lets you set an array as a property
- ☐ a property that lets you pass data to child elements

Reference

Q25. Which attribute is React's replacement for using innerHTML in the browser DOM?

- ☐ injectHTML
- ☑ dangerouslySetInnerHTML
- ☐ weirdSetInnerHTML
- ☐ strangeHTML

Reference

Q26. Which of these terms commonly describe React applications?

- ☑ declarative
- ☐ integrated
- ☐ closed
- ☐ imperative

Reference

Q27. When using webpack, why would you need to use a loader?

- ☐ to put together physical file folders
- ☑ to preprocess files
- ☐ to load external data
- ☐ to load the website into everyone's phone

Q28. A representation of a user interface that is kept in memory and is synced with the "real" DOM is called what?

- ☑ virtual DOM
- ☐ DOM
- ☐ virtual elements
- ☐ shadow DOM

Q29. You have written the following code but nothing is rendering. How do you fix this problem?

```
const Heading = () => {
  <h1>Hello!</h1>;
};
```

- ☐ Add a render function.
- ☑ Change the curly braces to parentheses or add a return statement before the `h1` tag.
- ☐ Move the `h1` to another component.
- ☐ Surround the `h1` in a `div`.

Q30. To create a constant in JavaScript, which keyword do you use?

- ☑ const
- ☐ let
- ☐ constant
- ☐ var

Q31. What do you call a React component that catches JavaScript errors anywhere in the child component tree?

- ☐ error bosses
- ☐ error catchers
- ☐ error helpers
- ☑ error boundaries

Reference

Q32. In which lifecycle method do you make requests for data in a class component?

- ☐ constructor
- ☑ componentDidMount
- ☐ componentWillReceiveProps
- ☐ componentWillMount

Reference

Q33. React components are composed to create a user interface. How are components composed?

- ☐ by putting them in the same file
- ☑ by nesting components
- ☐ with webpack
- ☐ with code splitting

Q34. All React components must act like _ with respect to their props.

- ☐ monads
- ☑ pure functions
- ☐ recursive functions
- ☐ higher-order functions

Q35. What is `[e.target.id]` called in this code snippet?

```
const handleChange = (e) => {
  setState((prevState) => ({ ...prevState, [e.target.id]: e.target.value }));
};
```

- ☐ a computed property name
- ☐ a set value
- ☑ a dynamic key
- ☐ a JSX code string

Reference

Q36. What is the name of this component?

```
class Clock extends React.Component {
  render() {
    return <h1>Look at the time: {time}</h1>;
  }
}
```

- ☑ Clock
- ☐ It does not have a name prop.
- ☐ React.Component
- ☐ Component

Q37. What is sent to an `Array.map()` function?

- ☑ a callback function that is called once for each element in the array
- ☐ the name of another array to iterate over
- ☐ the number of times you want to call the function
- ☐ a string describing what the function should do

Q38. Why is it a good idea to pass a function to `setState` instead of an object?

- ☐ It provides better encapsulation.
- ☐ It makes sure that the object is not mutated.
- ☐ It automatically updates a component.
- ☑ `setState` is asynchronous and might result in out of sync values.

Reference

Explanation: Because `this.props` and `this.state` may be updated asynchronously, you should not rely on their values for calculating the next state.

Q39. What package contains the render() function that renders a React element tree to the DOM?

- ☐ `React`
- ☑ `ReactDOM`
- ☐ `Render`
- ☐ `DOM`

Q40. How do you set a default value for an uncontrolled form field?

- ☐ Use the `value` property.
- ☑ Use the `defaultValue` property.
- ☐ Use the `default` property.
- ☐ It assigns one automatically.

Q41. What do you need to change about this code to get this code to run?

```
const clock = (props) => {
  return <h1>Look at the time: {props.time}</h1>;
};
```

- ☐ Add quotes around the return value
- ☐ Remove `this`
- ☐ Remove the render method
- ☑ Capitalize `clock`

Reference

Explanation: In JSX, lower-case tag names are considered to be HTML tags.

Q42. Which Hook could be used to update the document's title?

- ☑ `useEffect(function updateTitle() { document.title = name + ' ' + lastname; });`
- ☐ `useEffect(() => { title = name + ' ' + lastname; });`
- ☐ `useEffect(function updateTitle() { name + ' ' + lastname; });`
- ☐ `useEffect(function updateTitle() { title = name + ' ' + lastname; });`

Q43. Which function from React can you use to wrap Component imports to load them lazily?

- ☐ `fallback`
- ☐ `split`
- ☑ `lazy`
- ☐ `memo`

Reference

Q44. How do you invoke setDone only when component mounts, using hooks?

```
function MyComponent(props) {
  const [done, setDone] = useState(false);
  return <h1>Done: {done}</h1>;
}
```

☐ `useEffect(() => { setDone(true); });`

☑ `useEffect(() => { setDone(true); }, []);`

☐ `useEffect(() => { setDone(true); }, [setDone]);`

☐ `useEffect(() => { setDone(true); }, [done, setDone]);`

Q45. Currently, `handleClick` is being called instead of passed as a reference. How do you fix this?

```
<button onClick={this.handleClick()}>Click this</button>
```

☐ `<button onClick={this.handleClick.bind(handleClick)}>Click this</button>`

☐ `<button onClick={handleClick()}>Click this</button>`

☑ `<button onClick={this.handleClick}>Click this</button>`

☐ `<button onclick={this.handleClick}>Click this</button>`

Q46. Which answer best describes a function component?

☐ A function component is the same as a class component.

☑ A function component accepts a single props object and returns a React element.

☐ A function component is the only way to create a component.

☐ A function component is required to create a React component.

Q47. Which library does the `fetch()` function come from?

☐ FetchJS

☐ ReactDOM

☑ No library. `fetch()` is supported by most browsers.

☐ React

Q48. What will happen when this useEffect Hook is executed, assuming name is not already equal to John?

```
useEffect(() => {
  setName('John');
}, [name]);
```

☐ It will cause an error immediately.

☐ It will execute the code inside the function, but only after waiting to ensure that no other component is accessing the name variable.

☑ It will update the value of name once and not run again until name is changed from the outside.

☐ It will cause an infinite loop.

Q49. Which choice will not cause a React component to rerender?

☐ if the component calls `this.setState(...)`

☐ the value of one of the component's props changes

☐ if the component calls `this.forceUpdate()`

☑ one of the component's siblings rerenders

Q50. You have created a new method in a class component called handleClick, but it is not working. Which code is missing?

```
class Button extends React.Component{

  constructor(props) {
    super(props);
    // Missing line
  }

  handleClick() {...}
}
```

- ☐ `this.handleClick.bind(this);`
- ☐ `props.bind(handleClick);`
- ☐ `this.handleClick.bind();`
- ☑ `this.handleClick = this.handleClick.bind(this);`

Q51. React does not render two sibling elements unless they are wrapped in a fragment. Below is one way to render a fragment. What is the shorthand for this?

```
<React.Fragment>
  <h1>Our Staff</h1>
  <p>Our staff is available 9-5 to answer your questions</p>
</React.Fragment>
```

☐ A

```
<...>
  <h1>Our Staff</h1>
  <p>Our staff is available 9-5 to answer your questions</p>
</...>
```

☐ B

```
<//>
  <h1>Our Staff</h1>
  <p>Our staff is available 9-5 to answer your questions</p>
<///>
```

☑ C

```
<>
  <h1>Our Staff</h1>
  <p>Our staff is available 9-5 to answer your questions</p>
</>
```

☐ D

```
<Frag>
  <h1>Our Staff</h1>
  <p>Our staff is available 9-5 to answer your questions</p>
</Frag>
```

Q52. If you wanted to display the count state value in the component, what do you need to add to the curly braces in the `h1`?

```
class Ticker extends React.component {
  constructor(props) {
    super(props);
    this.state = { count: 0 };
  }
  render() {
    return <h1>{}</h1>;
  }
}
```

- ☑ this.state.count
- □ count
- □ state
- □ state.count

Q53. Per the following code, when is the Hello component assigned to greeting?

```
const greeting = isLoggedIn ? <Hello /> : null;
```

- □ never
- ☑ when `isLoggedIn` is true
- □ when a user logs in
- □ when the Hello function is called

Q54. In the following code block, what type is orderNumber?

```
ReactDOM.render(<Message orderNumber="16" />, document.getElementById('root'));
```

- ☑ string
- □ boolean
- □ object
- □ number

Q55. You have added a style property to the `h1` but there is an unexpected token error when it runs. How do you fix this?

```
const element = <h1 style={ backgroundColor: "blue" }>Hi</h1>;
```

- □ `const element = <h1 style="backgroundColor: "blue""}>Hi</h1>;`
- ☑ `const element = <h1 style={{backgroundColor: "blue"}}>Hi</h1>;`
- □ `const element = <h1 style={blue}>Hi</h1>;`
- □ `const element = <h1 style="blue">Hi</h1>;`

Q56. Which function is used to update state variables in a React class component?

- □ `replaceState`
- □ `refreshState`
- □ `updateState`
- ☑ `setState`

Q57. Consider the following component. What is the default color for the star?

```
const Star = ({ selected = false }) => <Icon color={selected ? 'red' : 'grey'} /
```

□ black

□ red

☑ grey

□ white

Q58. What is the difference between the click behaviors of these two buttons(assuming that this.handleClick is not bound correctly)

```
A. <button onClick=this.handleClick>Click Me</button>
B. <button onClick={event => this.handleClick(event)}>Click Me</button>
```

□ `Button A will not have access to the event object on click of the button`

☑ `Button A will not fire the handler this.handleClick successfully`

□ `There is no difference`

□ `Button B will not fire the handler this.handleClick successfully`

Q59. How would you add to this code, from React Router, to display a component called About?

☑ A

```
<Route path="/:id">
  {' '}
  <About />
</Route>
```

□ B

```
<Route path="/tid" about={Component} />
```

□ C

```
<Route path="/:id" route={About} />
```

□ D

```
<Route>
  <About path="/:id" />
</Route>
```

Reference

Q60. Which class-based component is equivalent to this function component?

```
const Greeting = ({ name }) => <h1>Hello {name}!</h1>;
```

□ A

```
class Greeting extends React.Component {
  constructor() {
    return <h1>Hello {this.props.name}!</h1>;
  }
}
```

□ B

```
class Greeting extends React.Component {
  <h1>Hello {this.props.name}!</h1>;
}
```

☑ C

```
class Greeting extends React.Component {
  render() {
    return <h1>Hello {this.props.name}!</h1>;
  }
}
```

☐ D

```
class Greeting extends React.Component {
  render({ name }) {
    return <h1>Hello {name}!</h1>;
  }
}
```

Q61. Give the code below, what does the second argument that is sent to the render function describe?

```
ReactDOM.render(
  <h1>Hi<h1>,
    document.getElementById('root')
)
```

- ☑ where the React element should be added to the DOM
- ☐ where to call the function
- ☐ where the root component is
- ☐ where to create a new JavaScript file

Q62. Why should you use React Router's Link component instead of a basic `<a>` tag in React?

- ☐ The link component allows the user to use the browser's `Back` button.
- ☐ There is no difference--the `Link` component is just another name for the `<a>` tag.
- ☐ The `<a>` tag will cause an error when used in React.
- ☑ The `<a>` tag triggers a full page reload, while the `Link` component does not.

Q63. What is the first argument, `x`, that is sent to the `createElement` function?

```
React.createElement(x, y, z);
```

- ☑ the element that should be created
- ☐ the order in which this element should be placed on the page
- ☐ the properties of the element
- ☐ data that should be displayed in the element

Q64. Which class-based lifecycle method would be called at the same time as this effect Hook?

```
useEffect(() => {
  // do things
}, []);
```

- □ componentWillUnmount
- ☑ componentDidMount
- □ render
- □ componentDidUpdate

Reference

Q65. What is the name of the base component of this component?

```
class Comp extends React.Component {
  render() {
    return <h1>Look at the time: {time}</h1>;
  }
}
```

- □ Comp
- □ h1
- □ React.Component
- ☑ Component

Q66. When using a portal, what is the first argument?

```
ReactDOM.createPortal(x, y);
```

- □ the current state
- ☑ the element to render
- □ the App component
- □ the page

Reference

Q67. What is `setCount`?

```
const [count, setCount] = useState(0);
```

- □ the initial state value
- □ a variable
- □ a state object
- ☑ a function to update the state

Reference

Q68. What is the use of map function below?

```
const database = [{ data: 1 }, { data: 2 }, { data: 3 }];
database.map((user) => <h1>{user.data}</h1>);
```

- □ gives a map of all the entries in database
- ☑ returns a heading tag for every entry in the database containing it's data
- □ returns one heading tag for all the entries in database
- □ checks which entry in the database is suitable for heading tag

Q69. Describe what is happening in this code?

```
const { name: firstName } = props;
```

- ☐ It is creating a new object that contains the same name property as the props object.
- ☐ It is assigning the value of the props object's firstName property to a constant called name.
- ☐ It is retrieving the value of props.name.firstName.
- ☑ It is assigning the value of the props object's name property to a constant called firstName.

Q70. What is wrong with this code?

```
const MyComponent = ({ names }) => (
  <h1>Hello</h1>
  <p>Hello again</p>
);
```

- ☐ React components cannot be defined using functions.
- ☑ React does not allow components to return more than one element.
- ☐ The component needs to use the return keyword.
- ☐ String literals must be surrounded by quotes.

Q71. When using a portal, what is the second argument?

```
ReactDOM.createPortal(x, y);
```

- ☐ the App component
- ☐ the page
- ☐ the current state
- ☑ the DOM element that exists outside of the parent component

Q72. Given this code, what will be printed in the `<div>` tag?

```
const MyComponent = ({ children }) => (
  <div>{children.length}</div>
);
...
<MyComponent>
<p>
  Hello <span>World!</span>
</p>
<p>Goodbye</p>
</MyComponent>
```

- ☐ It will produce an error saying "cannot read property "length" of undefined."
- ☐ 1
- ☐ undefined
- ☑ 2

Q73. What is this pattern called?

```
const [count, setCount] = useState(0);
```

☐ object destructuring
☑ array destructuring
☐ spread operating
☐ code pushing

Q74. What is the first file loaded by the browser in a basic React project?

☐ src/App.js
☐ src/index.js
☐ public/manifest.json
☑ public/index.html

Q75. The code below is rendering nothing and generates this error: "ReactDOM is not defined." How do you fix this issue?

```
import React from 'react';
import { createRoot } from 'reactjs-dom';

const element = <h1>Hi</h1>;
// Note: error on the line below
const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(element);
```

☑ `createRoot(document.getElementById("root"));`
☐ `ReactDOM(element, document.getElementById("root"));`
☐ `renderDOM(element, document.getElementById("root"));`
☐ `DOM(element, document.getElementById("root"));`

Q76. In this component, how do you display whether the user was logged in or not?

```
render() {
  const isLoggedIn = this.state.isLoggedIn;
  return (
    <div>
      The user is:
    </div>
  );
}
```

☐ `The user is loggedIn ? logged in : not logged in.`
☐ Write a function to check the login status.
☐ `The user is {isLoggedIn = "no"}.`
☑ `The user is {isLoggedIn ? "logged in." : "not logged in"}.`

Q77. You are rendering a list with React when this warning appears in the console: "Warning: Each child in a list should have a unique 'key' prop." How do you fix this issue?

☐ Add a key prop with the same value to each item in the list
☐ Clear the console warnings
☐ Use the UseId hook to generate a unique key for each element in the list
☑ When iterating over the list items, add a unique property to each list item.

Q78. How would you generate the boilerplate code for a new app that you are building to collect underpants?

- ☐ npm create-react-app collect-underpants
- ☐ npx start-app collect-underpants
- ☐ react new collect-underpants
- ☑ npx create-react-app collect-underpants

Source: React Docs

Q79. Add the code that will fire the photon torpedoes when the button is clicked.

```
class StarTrekkin extends React.Component {
  firePhotonTorpedoes(e) {
    console.log('pew pew');
  }
  render() {
    return; // Missing code
  }
}
```

- ☐ `<button onClick={firePhotonTorpedoes()}>Pew Pew</button>`
- ☐ `<button onClick={firePhotonTorpedoes}>Pew Pew</button>`
- ☐ `<button onClick={this.firePhotonTorpedoes()}>Pew Pew</button>`
- ☑ `<button onClick={this.firePhotonTorpedoes}>Pew Pew</button>`

Source: React Docs

Q80. What is the process of deciding whether an update is necessary?

- ☐ shadow DOM
- ☐ fiber
- ☑ reconciliation
- ☐ setting state

Q81. React is an open-source project but is maintained by which company?

- ☐ Intuit
- ☐ Twitter
- ☑ Facebook
- ☐ Snapchat

Q82. What command can you use to generate a React project?

- ☐ react-starter
- ☑ create-react-app
- ☐ react-gen
- ☐ react-start

hint

Q83. What is the browser extension called that React developers use to debug applications?

- ☑ React Developer Tools
- ☐ React Tooling Add-on
- ☐ React Codewatch
- ☐ React Debug

Q84. Which tool is not part of Create React App?

- ☐ React
- ☑ jQuery
- ☐ webpack
- ☐ ReactDOM

Q85. What is the JavaScript syntax extension that is commonly used to create React elements?

- ☐ HTML
- ☐ JavaScriptX
- ☑ JSX
- ☐ React JavaScript

Q86. How might you check property types without using Flow or TypeScript?

- ☐ Check Manually.
- ☐ Use `prop-helper` .
- ☑ use `prop-types` .
- ☐ user `checker-types` .

Q87. How do you add an id of heading to the following h1 element?

```
let dish = <h1>Mac and Cheese</h1>;
```

- ☐ `let dish = <h1 id={heading}>Mac and Cheese</h1>;`
- ☑ `let dish = <h1 id="heading">Mac and Cheese</h1>;`
- ☐ `let dish = <h1 id:"heading">Mac and Cheese</h1>;`
- ☐ `let dish = <h1 class="heading">Mac and Cheese</h1>;`

Q88. What value of button will allow you to pass the name of the person to be hugged?

```
class Huggable extends React.Component {
  hug(id) {
    console.log("hugging " + id);
  }
  render() {
    let name = "kitten";
    let button = // Missing code
    return button;
  }
}
```

- ☐ `<button onClick={(name) => this.hug(name)}>Hug Button</button>;`
- ☐ `<button onClick={this.hug(e, name)}>Hug Button</button>;`
- ☐ `<button onClick={(e) => hug(name, e)}>Hug Button</button>;`
- ☑ `<button onClick={(e) => this.hug(name, e)}>Hug Button</button>;`

Explanation: This question test knowledge of react class components. You need to use `this` in order to call methods declared inside class components.

Q89. What syntax do you use to create a component in React?

- ☐ a generator
- ☑ a function or a class

- ☐ a service worker
- ☐ a tag

Reference

Explanation: React Components are like functions that return HTML elements. Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML. Components come in two types, Class components and Function components.

Q90. You want to disable a button so that it does not emit any events onClick. Which prop do you use to acomplish this?

- ☐ onBlur
- ☐ onPress
- ☐ defaultValue
- ☑ disabled

Q91. In this function, which is the best way to describe the Dish component?

```
function Dish() {
  return (
    <>
      <Ingredient />
      <Ingredient />
    </>
  );
}
```

- ☐ child component
- ☑ parent component
- ☐ nested component
- ☐ sibling component

Q92. When does the componentDidMount function fire?

- ☑ right after the component is added to the DOM
- ☐ before the component is added to the DOM
- ☐ right after the component is updated
- ☐ right after an API call

Reference

Q93. What might you use webpack for in React development?

- ☐ to fetch remote dependencies used by your app
- ☑ to split your app into smaller chunks that can be more easily loaded by the browser
- ☐ to format your code so that it is more readable
- ☐ to ensure your app is not vulnerable to code injection

Q94. When using the React Developer Tools Chrome extension, what does it mean if the React icon is red?

- ☑ You are using the development build of React.
- ☐ You are using the production build of React.
- ☐ You are using webpack.
- ☐ You are using Create React App.

Reference

Q95. How would you modify the constructor to fix this error: "ReferenceError: Must call super constructor in derived class before accessing 'this'..."?

```
class TransIsBeautiful extends React.Component {
  constructor(props){
  // Missing line
  console.log(this) ;
  }
  ...
}
```

- ☐ render(props);
- ☑ super(props);
- ☐ super(this);
- ☐ this.super();

Q96. Which language can you not use with React?

- ☑ Swift.
- ☐ JSX.
- ☐ Javascipt.
- ☐ TypeScript.

Q97. This code is part of an app that collects Pokemon. How would you print the list of the ones collected so far?

```
constructor(props) {
    super(props);
    this.state = {
        pokeDex: []
    };
}
```

- ☐ console.log(props.pokeDex);
- ☐ console.log(this.props.pokeDex);
- ☐ console.log(pokeDex);
- ☑ console.log(this.state.pokeDex);

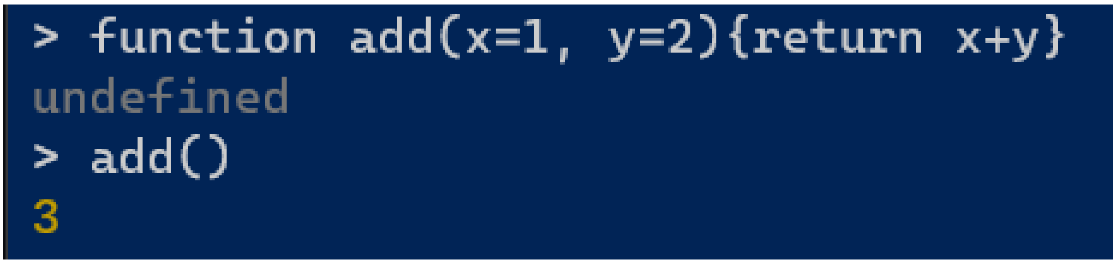
Reference

Q98. What would be the result of running this code?

```
function add(x = 1, y = 2) {
  return x + y;
}

add();
```

- ☐ null
- ☑ 3
- ☐ 0
- ☐ undefined



```
> function add(x=1, y=2){return x+y}
undefined
> add()
3
```

Explanation: function that called without parameter will use its param default value, thus x will always be default to 1 and y will always be default to 2.

Q99. Why might you use a React.ref?

- ☐ to refer to another JS file
- ☐ to bind the function
- ☐ to call a function
- ☑ to directly access the DOM node

Reference

Q100. What pattern is being used in this code block?

```
const { tree, lake } = nature;
```

- ☐ function defaults
- ☐ array destructuring
- ☐ PRPL pattern
- ☑ destructuring assignment

Reference

Q101. How would you correct this code block to make sure that the sent property is set to the Boolean value false?

```
ReactDom.render(
  <Message sent=false />,
  document.getElementById("root")
);
```

☑ A

```
<Message sent={false} />,
```

☐ B

```
ReactDom.render(<Message sent="false" />, document.getElementById('root'));
```

☐ C

```
<Message sent="false" />,
```

☐ D

```
ReactDom.render(<Message sent="false" />, document.getElementById('root'));
```

Passing Props to a Component

Q102. This code is part of an app that collects Pokemon. The useState hook below is a piece of state holding onto the names of the Pokemon collected so far. How would you access the collected Pokemon in state?

```
const PokeDex = (props) => {
  const [pokeDex, setPokeDex] = useState([]);
  /// ...
};
```

- ☐ props.pokeDex
- ☐ this.props.pokeDex
- ☐ setPokeDex()
- ☑ pokeDex

Explanation: useState always return an array with two values, the state itself (on first value) and the set function that lets you update the state (on second value) useState Reference

Q103. What would you pass to the onClick prop that will allow you to pass the initName prop into the greet handler?

```
const Greeting = ({ initName }) => {
  const greet = (name) => console.log("Hello, " + name + "!");
  return <button onClick={ ... }>Greeting Button </button>
}
```

- ☐ hug
- ☐ this.hug(initName)
- ☐ (name) => this.hug(name)
- ☑ () => hug(initName)

Explanation: Apparently the question misstyped `greet` as `hug`. Putting this aside, we can still learn from this question.

- In a function, the global object is the default binding for `this`. In a browser window the global object is [object Window]. This is a functional Component, so `this` from `this.hug` actually refers to browser window. Since it is a functional component, we can directly refer to hug without using `this`.
- To pass a handler to onClick, we should always pass a function rather than execute a function. So we need to use callback here. `initName` is available in Greeting's function scope, so we can directly supply as an argument to hug().

Q104. What is the name of the compiler used to transform JSX into JavaScript?

- ☑ Babel
- ☐ JSX Editor
- ☐ Browser Buddy
- ☐ ReactDOM

JSX Transform with Babel

Q105. Which hook is used to prevent a function from being recreated on every component render?

- ☑ useCallback
- ☐ useMemo
- ☐ useRef

- ☐ useTransition

React Hooks useCallback docuementation

Q106. Why might you use the `useRef` hook?

- ☐ To bind the function
- ☐ To call a function
- ☑ To directly access a DOM
- ☐ To refer to another JS file

Reference

Q107. Which of the following is required to use React?

- ☑ JavaScript
- ☐ React Router
- ☐ Redux
- ☐ Prop-Types

Reference

Q108. What is the correct way to get a value from context?

- ☐ const value = useContext(MyContext.Consumer)
- ☐ const value = useContext(MyContext.Provider)
- ☑ const value = useContext(MyContext)
- ☐ const value = useContext({value: "intiial value"})

Reference

Q109. Why is ref used?

- ☐ to bind function
- ☐ to call function
- ☑ to directly access DOM node
- ☐ to refer to another JS file

Reference

Q110. Choose the method which should be overridden to stop the component from updating?

- ☐ componentDidMount
- ☐ componentDidUpdate
- ☐ willComponentUpdate
- ☑ shouldComponentUpdate

Reference

Q111. What is the functionality of a “webpack” command?

- ☐ Runs react local development server
- ☐ Transfers all JS files to down into one file
- ☐ A module builder
- ☑ All of the above

Q112. Choose the method which is not a part of ReactDOM?

- ☐ ReactDOM.createPortal()
- ☐ ReactDOM.hydrate()
- ☑ ReactDOM.destroy()

☐ ReactDOM.findDOMnode()

Q113. In react, the key should be?

☑ Unique among his siblings
☐ Unique in DOM
☐ Does not requires to be unique
☐ all of the above

Reference

Q114. Which company developed ReactJS?

☐ Google
☑ Meta (ex Facebook)
☐ Apple
☐ Twitter

Reference

Q115. Choose the library which is most often associated with react?

☐ Chai
☐ Sinon
☑ Jest
☐ Mocha

Reference

Q116. What of the following is used in React.js to increase performance?

☐ Original DOM
☑ Virtual DOM
☐ Both of the above
☐ None of the above

Reference

Q117. Among The following options, choose the one which helps react for keeping their data uni-directional?

☐ DOM
☑ flux
☐ JSX
☐ Props

Reference

Q118. Which choice is a correct refactor of the Greeting class component into a function component?

```
class Greeting extends React.Component {
  render() {
    return <h1>Hello {this.props.name}!<h1>;
  }
}
```

☐ `const Greeting = (name) => <h1>{name}</h1>`
☐ `function Greeting(name){return <h1>{name}</h1>;}`
☐ `const Greeting = props => { <h1>{props.name}</h1> }`
☑ `const Greeting = ({ name }) => <h1>Hello {name}</h1>;`

Q119. Why is the `waitlist` not updating correctly?

```
const Waitlist = () => {
  const [name, setName] = useState('');
  const [waitlist, setWaitlist] = useState([]);
  const onSubmit = (e) => {
    e.preventDefault();
    waitlist.push(name);
  };
  return (
    <div>
      <form onSubmit={onSubmit}>
        <label>
          Name: <input type="text" value={name} onChange={(e) => setName(e.target
        </label>
        <button type="submit">Add to waitlist</button>
      </form>

      <ol>
        {waitlist.map((name) => (
          <li key={name}>{name}</li>
        ))}
      </ol>
    </div>
  );
};
```

- ☑ `waitlist` is being mutated directly. Use the `setWaitlist` function instead to update the waitlist state.
- □ The form is reloading the page each time `Add to waitlist` is clicked.
- □ The `Add to waitlist` button is missing a click handler.
- □ There are likely repeated names inside of the `waitlist` array.

Reference

Q120. What is the pattern that is used in the Context.Consumer below?

<Context.Consumer> {(isLoggedIn)=>**{isLoggedIn ? "Online" : "Offline"}**}
</Context.Consumer>

- □ higher-order component
- □ wish component
- ☑ Render Props
- □ setup Componet

Q121. In React.js which one of the following is used to create a class for Inheritance ?

- □ Create
- ☑ Extends
- □ Inherits
- □ Delete

Reference

Q122. What is the purpose of render() in React.js?

- □ To replace the existing markup
- □ To update the existing markup
- ☑ Both of the above
- □ None of the above

Reference

Q123. What is the use of super(props) in React.js?

- □ To call the constructor of the parent class
- □ To initialize this.props in the constructor
- ☑ Both of the above
- □ None of the above

Reference

Q124. What is Redux in React.js?

- ☑ A state container for JavaScript apps
- □ A tool for building UI components
- □ Both of the above
- □ None of the above

Reference

Q125. What is the purpose of the virtual DOM in React.js, and how does it improve performance in web applications??

- □ The virtual DOM is used to store user authentication data securely.
- □ The virtual DOM is a backup copy of the actual DOM, created for debugging purposes.
- ☑ The virtual DOM is a lightweight representation of the actual DOM, and it helps improve performance by minimizing direct manipulation of the real DOM.
- □ The virtual DOM is a database used to store component states.

Reference

Q126. You run the following code and get this error message: "invalid hook call." what is wrong with the code?

```
import React from 'react';

const [poked, setPoked] = React.useState(false);

function PokeButton() {
  return <button onClick={() => setPoked(true)}>{poked ? 'You have left a poke.'
}
```

- ☑ The useState call needs to be called inside of the PokeButton component.
- □ The react package is likely not installed correctly.
- □ useState is not imported correctly. Import useState directly instead of importing react.
- □ PokeButton is a pure function and therefore cannot have any local state.

Q127. A colleague comes to you for help on a react component. They say that the poke button renders correctly, however when the button is clicked, this error is shown:

"setPoked is not defined". What is wrong with their code?

```
function PokeButton() {
  const { poked, setPoked } = useState(false);
  return <button onclick={() => setPoked(true)}>{poked ? 'You have left a poke.'
}
```

- ☐ onClick prop should be onclick.
- ☐ The click handler passed to the onClick prop is inlined. Move this handler into a variable outside of JSX.
- ☑ They use object destructructing instead of array destructructing. Wrap the poked and setPoked values in an array.
- ☐ poked and setPoked are not destructured in the correct order.

Q128. This component is loaded dynamically. What should you replace XXXX with to complete the code?

```
const OtherComponent = React.lazy(() => import('./OtherComponent.js'));

function MyComponent() {
  return (
    <XXXX fallback={<spinner />}>
      <OtherComponent />
    </XXXX>
  );
}
```

- ☐ Component
- ☐ Fragment
- ☑ Suspense
- ☐ Lazy

Q129. Elements in lists in React should have __ that are ___ .

- ☑ keys ; unique
- ☐ keys ; indexes
- ☐ style ; inline
- ☐ values ; not-null Source: React Docs

Q130. You want to memorize a callback function so you ensure that React does not recreate the function at each render. Which hook would you use to accomplish this?

- ☐ useRef
- ☐ useMemo
- ☐ memo
- ☑ useCallback

Source: CodeDamn

Q131. You want to perform a network operation as the result of a change to a component's state named userInput. what would you replace XXXX with?

```
useEffect(callNetworkFunc, XXXX);
```

- ☑ [userInput]
- ☐ userInput
- ☐ undefined
- ☐ []

Q132. When is the Hello component displayed?

```
<div>{isLoggedIn ? <Hello /> : null}</div>
```

- ☐ when isLoggedIn is false
- ☑ when isLoggedIn is true
- ☐ when isLoggedIn is false and the Hello function is invoked
- ☐ never

Q133. When do you use `useLayoutEffect`?

- ☐ to optimize for all devices
- ☐ to complete the update
- ☑ to change the layout of the screen
- ☐ when you need the browser to paint before the effect runs

Q134. What is the difference between state and props in React?

- ☐ Props are set by the parent component, state is set by the child component
- ☑ Props are passed to a component, state is managed within the component
- ☐ Props can be updated, state cannot be updated
- ☐ There is no difference - props and state are the same

Q135. Which language can you not use with React?

- ☑ Swift.
- ☐ JSX.
- ☐ Javascipt.
- ☐ TypeScript.

Q136. Which answer best describes a function component?

- ☐ A function component is the same as a class component.
- ☑ A function component accepts a single props object and returns a React element.
- ☐ A function component is the only way to create a component.
- ☐ A function component is required to create a React component.

Q137. Which library does the `fetch()` function come from?

- ☐ FetchJS
- ☐ ReactDOM
- ☑ No library. `fetch()` is supported by most browsers.
- ☐ React

Q138. In React, what is the purpose of the key prop when rendering a list of components

- ☐ The `key` prop is used to provide a unique identifier for the component.

- □ The `key` prop is used to define the color of the component.
- □ The `key` prop is required to render a list of components.
- ☑ The `key` prop is used by React to optimize updates and identify which items have changed or been added/removed in the list.

Q139. What is the primary function of React Router?

- □ React Router is used for fetching data from APIs.
- □ React Router is used to create animations in React applications.
- □ React Router is used for managing state in React components.
- ☑ React Router is used for adding navigation and routing to React applications, allowing users to navigate between different views or pages.

Q140. When should you use Redux in a React application?

- □ Redux is always required in React applications.
- □ Redux should be used when you need to fetch data from APIs.
- □ Redux is used for creating user interfaces but not for state management.
- ☑ Redux is typically used when you have complex state management needs, such as sharing state between multiple components or handling deeply nested state.

Q141. What is the use of React hooks?

- □ To optimize React apps for mobile devices
- □ To add visual effects to React components.
- ☑ To allow using state and lifecycle methods in function components
- □ To integrate with external UI libraries like Bootstrap

Q142. How can you pass data through a React component tree without having to pass props down manually at every level?

- ☑ By using React context
- □ By using redux
- □ By using react router
- □ By using react lifecycle methods