**Java Script**

#### 1. When would the final statement in the code shown be logged to the console?

let modal = document.querySelector('#result');

setTimeout(function(){

modal.classList.remove('hidden);

}, 10000);

console.log('Results shown');

* after 10 second
* after results are received from the HTTP request
* after 10000 seconds
* immediately

#### 2. Which statement creates a new function called discountPrice?

* [ ]

let discountPrice = function (price) {

return price \* 0.85;

};

* [ ]

let discountPrice(price) {

return price \* 0.85;

};

* [ ]

let function = discountPrice(price) {

return price \* 0.85;

};

* [ ]

discountPrice = function (price) {

return price \* 0.85;

};

#### 3. What is the result in the console of running the code shown?

var Storm = function () {};

Storm.prototype.precip = 'rain';

var WinterStorm = function () {};

WinterStorm.prototype = new Storm();

WinterStorm.prototype.precip = 'snow';

var bob = new WinterStorm();

console.log(bob.precip);

* Storm()
* undefined
* 'rain'
* 'snow'

#### 4. What is the result in the console of running this code?

'use strict';

function logThis() {

this.desc = 'logger';

console.log(this);

}

new logThis();

object created from logThis function-constructtor {desc: 'logger'}, inside function-constructor ***this*** is always a newly created object

#### 5. What does the following expression evaluate to?

[] == [];

* True
* undefined
* []
* False

#### 6. What will be logged to the console?

var a = ['dog', 'cat', 'hen'];

a[100] = 'fox';

console.log(a.length); // 101

#### 7. What will be the output of this code?

const arr1 = [2, 4, 6];

const arr2 = [3, 5, 7];

console.log([...arr1, ...arr2]);

* [2, 3, 4, 5, 6, 7]
* [3,5,7,2,4,6]
* [3, 5, 7, 2, 4, 6]
* [[2, 4, 6], [3, 5, 7]]
* [2, 4, 6, 3, 5, 7]

**8. What is the output of this code?**

let x = 6 + 3 + '3';

console.log(x);

* 93
* 12
* 66
* 633

**9. What is the output of this code?**

let rainForests = ['Amazon', 'Borneo', 'Cerrado', 'Congo'];

rainForests.splice(0, 2);

console.log(rainForests);

* ["Amazon","Borneo","Cerrado","Congo"]
* ["Cerrado", "Congo"]
* ["Congo"]
* ["Amazon","Borneo"]

#### 10. 0 && hi

* ReferenceError
* True
* 0
* False

#### 11. Which Variable-defining keyword allows its variable to be accessed (as undefined) before the line that defines it?

* all of them
* const
* var
* let

#### 12. Which of the following values is not a Boolean false?

* Boolean(0)
* Boolean("")
* Boolean(NaN)
* Boolean("false")

#### 13. Which of the following is not a keyword in JavaScript?

* this
* catch
* function
* array

#### 15. What is the result of running this code?

sum(10, 20);

diff(10, 20);

function sum(x, y) {

return x + y;

}

let diff = function (x, y) {

return x - y;

};

* 30, ReferenceError, 30, -10
* 30, ReferenceError
* 30, -10
* ReferenceError, -10

**React**

**1. 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

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

const [, , animal] = ['Horse', 'Mouse', 'Cat'];

console.log(animal);

* Horse
* Cat
* Mouse
* Undefined

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

* JSX Editor
* ReactDOM
* Browser Buddy
* Babel

#### 4. What can you use to handle code splitting?

* React.memo
* React.split
* React.lazy
* React.fallback

#### 5. 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

#### 6. 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.

#### 7. Which attribute do you use to replace innerHTML in the browser DOM?

* injectHTML
* dangerouslySetInnerHTML
* weirdSetInnerHTML
* strangeHTML

#### 8. 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

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

* constructor
* componentDidMount
* componentWillReceiveProps
* componentWillMount

#### 10. Why might you use a ref?

* to directly access the DOM node
* to refer to another JS file
* to call a function
* to bind the function

#### 11. 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]);

#### 12. Which of the following click event handlers 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 = "kitteh";

let button = <button onClick={() => {

this.hug(name);

}}>Click Me</button>;

return button;

}

}

#### 13. 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>;

#### 14. 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.

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

useEffect(() => {

// do things

}, []);

* componentWillUnmount
* componentDidMount
* render
* componentDidUpdate