

Technical Test

JavaScript

June 2022

Presented by
Tomás Rodeghiero

JAVASCRIPT EXERCISES

1. Reading the following statements, please indicate which information will be printed on the screen.

```
<html>
<script type="text/javascript">
  var numLlamadas = 0;

  function calculaBinomio(a, b)
  {
    var cuadradoDeA = a*a;
    var cuadradoDeB = b*b;
    numLlamadas++;

    return cuadradoDeA + cuadradoDeB + 2*a*b;
  }

  alert(calculaBinomio(2, 3));
  alert(calculaBinomio(4, 5));
  alert(numLlamadas);
  alert(typeof(cuadradoDeA));

</script>
</html>
```

* Explain in detail the 4th output:

```
alert(typeof(cuadradoDeA));
```

ANWER

Based on the function **calculaBinomio(a, b)**, the given calls return the following results:

- **alert(calculaBinomio(2, 3));** <- this will return: **25**.
- **alert(calculaBinomio(4, 5));** <- this will return: **81**.
- **alert(numLlamadas);** <- this will return: **NaN** (*Not a Number*), because this function receives two values as parameters and in this case it is invoked with only parameter.
- **alert(typeof(cuadradoDeA));** <- this will return: **undefined**, because we want to print on the screen by means of an alert the data type of “cuadradoDeA”, but “cuadradoDeA” is not defined as a global variable of the program, but as a local variable of the present function, for this reason we do not have a value for it in this context.

As an annex, we could mention that if we had invoked an alert in the following way “**alert(typeof(25))**”, then we would have obtained on the screen that 25 is a number.

2. As you probably know in JavaScript there is the variable type String. One of its functions is “toUpperCase”, therefore we can do the following thing:

```
var s2 = "Hello World!";
alert(s2.toUpperCase ()); <- this will return: HELLO WORLD!
```

How do you rewrite the above function in order to return this text (HELLO WORLD!) in small letters?

ANSWER

To return this text in lowercase make use of the JavaScript predefined function “toLocaleLowerCase”. For this reason, to get “Hello World!” in lowercase based on the given code, we should write the following.

```
var s2 = "Hello World!";
alert(s2.toLocaleLowerCase ()); <- this will return: hello world!
```

3. Using the next function, the console should show the message: “Openbravo!!!”

```
mostrar = function(param){
    param();
}
```

ANSWER

Using only the data of the presented function, without altering any line of code, for this function to display the message “Openbravo!!!” by console, we could write the following code.

```
mostrar = function(param) {
    function param() {
        console.log(“Openbravo!!!”);
    }
}
mostrar();
```

4. Write a JavaScript program to check whether a matrix is a diagonal matrix or not. In linear algebra, a diagonal matrix is a matrix in which the entries outside the main diagonal are all zero (the diagonal from the upper left to the lower right).

Example:

```
[1, 0, 0], [0, 2, 0], [0, 0, 3] ] = true
[1, 0, 0], [0, 2, 3], [0, 0, 3] ] = false
```

ANWER

In order to determine whether a matrix is diagonal or not, we could write the following function, which after several tests has been verified.

```
IsDiagonalMatrix = function diagonalMatrix(matrix) {
  for (let i = 0; i < matrix.length; i++) {
    for (let j = 0; j < matrix.length; j++) {
      if (i !== j && matrix[i][j] !== 0) {
        return false;
      }
    }
  }
  return true;
}
```

REACT EXERCISES

1. What are props in a component?

Select all the options that apply.

- ☐ Are the component parameters
- ☐ is the state of the component
- ☒ It is a mechanism to put data inside the component

2. What is a component?

Mark only one option.

- ☐ An element
- ☒ A function or class that returns an element
- ☐ An element that returns a function or class
- ☐ Other: _____

3. Which of these component types are valid?

Select all the options that apply.

- ☒ Complete Component
- ☒ Class Component
- ☒ Component Function
- ☐ Parent Component
- ☐ Child Component

4. What is the state of a component?

ANSWER

The **state of a component** in React JS refers to the internal values that manage the logic and data of a component, which allow it to react to any change that will cause the interface to be render again.

It should be noted that the state of a component is not accessible from another component except from the one that owns and assigns it.

5. What are the component lifecycle methods

ANSWER

The **component lifecycle methods** are procedures that are automatically executed in a Class Component, which occur in 3 (three) phases:

- i. Assembly.
- ii. Update.
- iii. Disassembly.

6. How would you set the state of a component?

ANSWER

The **state of a component** is set by using the `setState()` method.

7. Which of these implementations change the state of a component?
Check only one option.

ANSWER

- `this.state.value = "my value"`
- ✓ ● `this.setState({ value: "my value" })`
- `this.state = { "value": "my value" }`
- Others: _____

8. What happens when we change the state of a component?

ANSWER

When we **change the state of a component** it happens that each component will react to internal changes, caused in its state, or external, based on new properties received, and will be re-rendered in the interface.

9. Write a component that renders a list of elements, given a prop that contains an array of objects in the following form: { firstname: "demo", lastname: "demo", dni: 1234 }

ANSWER

Next, we will write a component that displays a list of elements, given a prop as stated in the statement.

```
const Person = (prop) => {
  return (
    <div>
      <h2>{prop.firstname} </h2>
      <h1>{prop.lastname}</h1> { /* It is a h1 because I consider that
                                the last name serves to better identify
                                a person rather than his name */ }
      <p>{prop.dni}</p>
    </div>
  )
}
```

10. Write a component that receives a prop called "data", and renders an H1 with the content of data.

ANSWER

Next, we will write a component that receives a prop called "data", and renders an H1 with the content of data.

```
function renderH1(data) {
  return (
    <h1>{data}</h1>
  );
}
```

11. How do you import Javascript modules installed with NPM or YARN? Check only one option.

ANSWER

- ☒ import name from 'node_modules/name';
- ☐ import name from '../node_modules/name/name.js'
- ☐ import name from '../node_modules/name/index.js'
- ☐ import name from 'name';
- ☐ Others: _____

12. How do I import the following component, considering that it is inside a file in the same directory, called MyComponent.js?

```
export default class MiComponente extends React.Component {  
  render() {  
    return (  
      <h1>Hello world</h1>  
    )  
  }  
}
```

Mark only one option.

- ☒ • import MyComponent from './MyComponent';
- import { MyComponent } from 'MyComponent';
- import MyComponent from 'MyComponent.js';
- import { MyComponent } from './MyComponent.js';
- Others: _____

13. What is the error in the following component?

```
const MiComponente = () => {  
  <h1>  
    Hello World  
  </h1>  
}
```

ANSWER

The **error** in the given component is that it returns nothing, or in other words, it returns a void, and for this component to be correct it should return a JSX element.

Attached below is a code that allows to correct "MiComponente".

```
const MiComponente = () => {  
  return <h1>Hola Mundo</h1>;  
};
```

14. Explain, in your own words, what this component does.

```
export class Componente extends React.Component {

  constructor(props) {
    super(props);
    this.state = {
      property: 1
    }
  }

  click = (propertyValue) => {
    this.setState({ property: propertyValue + 1 })
  }

  render() {
    return (
      <>
        <h1>{this.state.property}</h1>
        <button onClick={() => this.click(this.state.property)}>
          Click here
        </button>
      </>
    )
  }
}
```

ANWER

What this component does is to render a button based on a certain constructor and variable that have been created.

15. What errors do you see in the following component?

```
class Componente extends React.Component {
  render() {
    this.setState(property: 'Propiedad')
    return (
      <h1>El estado es: {this.state.property}</h1>
    )
  }
}
```

ANWER

The errors made in this component is that the state of a component should not be modified inside a render function and, at the same time, setState should be enclosed in square brackets because as it is declared we would have another error.

16. How would you make a Rest API call from JavaScript?

ANSWER

Personally, I would make a Rest API call from JavaScript using the **Fetch API**, although there are also other ways to do it such as AJAX and JQuery, among others.

17. What is the use of the useEffect hook?

ANSWER

The **useEffect** hook is used to make use of the life cycle in a functional component, which receives as parameter a function that will be executed every time our component is rendered, either by changes in the state or properties. In addition, with useEffect we can also subscribe and unsubscribe to events, timers, services, API's, etc.

18. How would you develop a custom hook that exposes methods to save and modify a counter? The hook must have a parameter that allows the developer to specify how much the counter should add for each call to the setter.

ANSWER

Import React, { useState, useEffect } from 'react';

```
const App = (count) => {
  const [count, setCount] = useState(0)
  return (
    setCount(count + count)
  );
};
```

Note: I admit that I did not know how to solve this exercise, so I promise to improve my understanding in the use of React JS hooks.

FRP
Etendo