**Parte 2**

En esta parte evaluaremos la lógica en implementación de funciones para resolver problemas específicos. Por favor intentar resolver los problemas en la menor cantidad de líneas de código posible y con la mayor eficiencia en ejecución. Los problemas se deben resolver en Javascript, no se aceptarán soluciones en otros lenguajes. Para la entrega de esta parte, por favor entregar un archivo llamado ‘parte2’ en el archivo .zip de la entrega general.

1. **Escribir una función que recibe un arreglo de strings y retornar un arreglo con la longitud de cada string.**

function getLenght(strings) {

var length = [];

for (var i = 0; i < strings.length; i++) {

length.push(strings[i].length);

}

return length;

}

var strings = ["Hola", "mi", "nombre", "es", "Juliana"];

var length = getLenght(strings);

console.log(length);

**2. Escribir una función que encuentre un número en un arreglo de números de la forma más eficiente posible. El arreglo de números de entrada puede estar desordenado.**

function findNumber(arr, number) {

return arr.indexOf(number);

}

var numbers = [3, 7, 1, 9, 4];

var lookNumber = 4;

var index = findNumber(numbers, lookNumber);

console.log("El número", lookNumber, "se encuentra en el índice", index);

**3. Escribir una función que al ser llamada saludo(‘Hola’)(‘Mundo’) imprima el texto ‘Hola Mundo’. Ojo, la función se debe ejecutar tal cual saludo(‘Hola’)(‘Mundo’), no hay error al escribir este enunciado.**

function greeting(word1) {

return function (word2) {

console.log(word1 + " " + word2);

};

}

saludo("Hola")("Mundo");

**4. Explique qué significa en sus propias palabras: Function.prototype.bind**

Es un método del objeto Function que permite establecer y manipular el valor de *this* en una función con la posibilidad de fijar argumentos, de esta manera nos permite crear una copias de funciones existentes que ya tiene un valor previamente predefinido, así cuando se llame la función se está creando una copia y será más fácil ejecutar la función incluso si la llamamos desde otro lugar. Cuando vamos a ejecutar una función creamos una nueva función donde podemos manipular el valor this y/o los parámetros de la función. lo que facilita su reutilización y simplifica la llamada a la función en diferentes contextos.

**5. ¿Qué es la coerción en Javascript?**

Es la conversión automática o implicita de valores de un tipo de dato a otro durante una operación (Ejemplo: número a boolean, string a número, número a string). También usando funciones o métodos específicos. Por ejemplo, parseInt() o parseFloat()o el operador +

**6. ¿Qué es una IIFE (Immediately Invoked Function Execution)? Escribe un ejemplo.**

Es un tipo de función que se invoca inmediatamente después de que la definimos de esta manera podemos encapsular código dentro de un ámbito local y evitar la contaminación del ámbito global y/o la privacidad de datos.

(function() {

var msg = "Hola, soy una función!";

console.log(msg);

})();

**7. ¿Cuál es la diferencia entre una variable que es: null, undefined o undeclared? ¿Cómo se verifica cada uno de estos estados?**

Son tres estados diferentes para las variables, cada uno de estos estados representa una situación diferente.

1. **null**: indica la ausencia de valor o la intención de asignar un valor vacío a una variable.
2. **undefined**: es un valor que se asigna automáticamente a una variable que ha sido declarada pero no ha sido inicializada con ningún valor o es el valor de retorno de una función si no se especifica un valor de retorno
3. **undeclared**: es un valor que significa que la variable no ha sido declarada. Esto ocurre cuando intentamos acceder a una variable que no ha sido declarada con var, let o const
4. **¿Cómo se verifica cada uno de estos estados?**

* Para verificar si es null, podemos usar el operador de igualdad estricta (===) para compararla con null.
* var variable = null;
* if (variable === null) {
* console.log("La variable es null");
* }
* Para verificar si es undefined, podemos usar el operador de igualdad estricta (===) para compararla con undefined.

var variable;

if (variable === undefined) {

console.log("La variable es undefined");

}

* No hay una forma exacta para verificar si una variable es undeclared ya que se genera un error cuando intentamos acceder a una variable no declarada.

*// No se declara la variable “variable"*

console.log(variable);

*// Lanzará un error de referencia: "variable is not defined"*

**8. ¿Cuáles son los principales componentes de Angular?**

Los principales componentes de Angular son:

1. **Módulo:** Son contenedores lógicos que agrupan componentes, servicios y herramientas que usamos para organizar y configurar la aplicación.
2. **Componente:** Son la base de Angular, son elementos visuales y funcionales de la interfaz de usuario. Cada componente tiene una plantilla, una clase para definir su comportamiento o funciones y los archivos de estilos.
3. **Directiva:** Son instrucciones para modificar el comportamientos o apariencia de un elementos del DOM, hay directivas prediseñadas por ejemplo ngIf o ngFor pero también podemos personalizarlas.
4. **Servicio:** Clases que proporcionan funcionalidad que puedo usar en diferentes componente en toda la aplicación siendo así más fácil la reutilización del código.
5. **Enrutamiento**: Es un enrutador integrado que permite la navegación de vistas o componentes de la aplicación.
6. **Formularios**: conjunto de herramientas para trabajar con formularios, incluyendo enlace de datos bidireccional, validación y seguimiento de estado de los campos.

También incluye otras características y herramientas, como manejo de eventos, pipes, testing, guards, entre otros, que permiten desarrollar aplicaciones web robustas y escalables.

**9. ¿Qué es un JSX?**

Es una combinación entre JavaScript y XML/HTML es un solo archivo permitiendo que la estructura y apariencia de los elementos dentro de la interfaz de usuario más eficiente. Con JSX podemos escribir elementos HTML directamente en nuestro código JavaScript mezclando la lógica y los elementos de la interfaz en un solo lugar.

**10. ¿Qué es el state en Angular? Que hace el método render?**

El estado en Angular se refiere a los datos y propiedades que un componente utiliza para determinar su comportamiento y apariencia. El método render se encarga de actualizar la vista del componente en respuesta a los cambios en el estado, asegurando que la interfaz de usuario refleje de manera precisa y actualizada la información y el estado del componente.

**11. ¿Cómo funciona el router en Angular?**

Es una funcionalidad que permite la navegación entre diferentes vistas o componentes en una sola página. Podemos definir rutas y asociarlas a componentes específicos que permiten cargar y mostrar los componentes cuando el usuario va navegando por diferentes rutas de la aplicación. Permite definir rutas, asociarlas con componentes específicos y cargar los componentes correspondientes cuando el usuario navega por la aplicación.