@autor: Victor Manuel Resendiz Cortina

Ejercicios:

1. Tiene dos arreglos de String, ‘clientes’ y ‘empleados’, y deseas combinarlas para crear un arreglo de contactos. ¿Qué método sería el más adecuado para esta tarea? Prueba todas las opciones y describe el resultado.

Para facilitar la salida en un solo arreglo, la opción viable es concat

a. Concat

```
// El metodo concat fusionar matrices, es decir no modifica lass matrices,
// si no que devuelvve una nueva matriz
let clientes =['cliente1','cliente2','cliente3' ];
let empleados = ['empleado1', 'empleado2','empleado3'];
let contactosConcat = empleados.concat(clientes);
console.log(contactosConcat);
```

b. Join

```
// El metodo join regresa un arreglo uniendo ambos arreglos, es decir
// guarda el primer arreglo y seguido de eso agrega los elementos del
// segundo arreglo
clientes =['cliente1','cliente2','cliente3' ];
empleados = ['empleado1', 'empleado2','empleado3'];
let contactosJoin = empleados.join(`- ${clientes} -`);
console.log(contactosJoin);
```

c. Push

```
// El metodo push regresa un arreglo guardando los elementos del primer arreglo enseguida
// coloca el arreglo en la ultima posicion
clientes =['cliente1','cliente2','cliente3' ];
empleados = ['empleado1', 'empleado2','empleado3'];
let contactosPush = empleados.push(clientes);
console.log(contactosPush);
console.log(empleados);
```

d. Splice

```
// El metodo splice regresa un nuevo arreglo modificando, eliminando o agregando elementos
// en este caso se agrega el arreglo en la ultima posicion
clientes =['cliente1','cliente2','cliente3' ];
empleados = ['empleado1', 'empleado2','empleado3'];
let contactosSplice = empleados.splice(1,0,clientes);
console.log(contactosSplice);
console.log(empleados);
```

@autor: Victor Manuel Resendiz Cortina

2. Teniendo el siguiente arreglo:

var numbers = [5, 32, 43, 4];

Y la siguiente función:

numbers.filter(function(n) { return n % 2 !== 0; });

Agrega una variable para alojar el resultado de la función filter() y explica cual es el resultado.

```
var numbers = [5,32,43,4];

let nfilter = numbers.filter(function (n) {return n%2 !==0});
console.log(nfilter);
// Se crea lavariable nFilter, la cual guarda el resultado de la expresion,
// al usar el metodo filter(), se recorre el arreglo y la funcion
// extrae los elementos que no sean pares
```

3. Dado el siguiente arreglo de objetos:

```
var people = [{
        id: 1,
        name: "John",
        age: 28
}, {
        id: 2,
        name: "Jane",
        age: 31
}, {
        id: 3,
        name: "Peter",
        age: 55
}];
```

Genera la función en javascript que filtre a las personas menores de 35 años.

```
//se manda a llamar la funcion, pasando como argumento el arreglo
edadMenor35(people);

// se declara la funcion, y se realiza un recorrido al arreglo obtenido para conocer
// la longitud y en base a eso hacer una comparacion si la edad es menor a 35
// se imprime el nombre de la persona
function edadMenor35(params) {
    for (let i = 0; i < params.length; i++) {
        const element = params[i];
        if (element.age<35) {
            console.log(element.name);
        }
    }
}
```

@autor: Victor Manuel Resendiz Cortina

4. Dado el siguiente arreglo de objetos:

```
let people = [
        { name: "bob", id=1 }, { name: "john", id=2 },
        { name: "alex", id=3 }, { name: "john", id=3 }
];
```

Genera la función para obtener el numero de veces que aparece cada nombre.

```
// Se llama la función
numVecesNombre(people)

// Se le pasa el arreglo de objeto a la funcion
// la cual realiza un recorrido, obteniendo los elementos,
// y estos a su vez nos permiten obtener el nombre
// el cual se guarda en un arreglo para recorrerlo
// y obtener el numero de veces que aparece un nombre
// en todo el arreglo
function numVecesNombre(params) {
    let nombresRepetidos=[];
    for (let i = 0; i < params.length; i++) {
        const element = params[i];
        nombresRepetidos.push(element.name);
    }
    let result={};
    nombresRepetidos.forEach((value)=>{
        result[value]=(result[value]||0)+1;
    });
    console.log(result);
}
```

5. Dado el siguiente arreglo:

```
var myArray = [1, 2, 3, 4];
```

Genera la función para imprimir en consola el numero máximo y el numero mínimo.

@autor: Victor Manuel Resendiz Cortina

```
// se llama a la funcion menor
menor(myArray);

// se realiza de 2 formas, recorriendo el array y
// y haciendo una condicional en la cual el numero menor
// se guarda en una variable la cual se imprime al final
// la segunda forma es usando em metodo Math.min
function menor(myArray) {
    // for (let i = 0; i < myArray.length; i++) {
    //     if (myArray[i]<min) {
    //         min=myArray[i]
    //     }
    // }
    // console.log(`el valor minimo es ${min}`);
    let min = Math.min(...myArray);
    console.log(min);
}
```

```
// se llama a la funcion mayor
mayor(myArray);
// se realiza de 2 formas, recorriendo el array y
// y haciendo una condicional en la cual el numero mayor
// se guarda en una variable la cual se imprime al final
// la segunda forma es usando em metodo Math.max
function mayor(myArray) {
    // for (let i = 0; i < myArray.length; i++) {
    //     if (myArray[i]>min) {
    //         min=myArray[i]
    //     }
    // }
    // console.log(`el valor minimo es ${min}`);
    let min = Math.max(...myArray);
    console.log(min);
}
```

@autor: Victor Manuel Resendiz Cortina

6. Teniendo el siguiente objeto:

```
var object = {
        key1: 10,
        key2: 3,
        key3: 40,
        key4: 20
};
```

Generar la función para pasar cada elemento ‘key’ a un arreglo y ordenarlos según su valor.

```
// se crea una variable la cual guarda el resultado
// el cual contiene el valor de cada lemento ordenado
// de acuerdo al metodo sort()
console.log(object);
let acomodo = Object.values(object).sort((a,b)=>a-b);
console.log(acomodo);
```

7.- Teniendo el siguiente arreglo:

```
var personArr=[
    {
        "personId":123,
        "name":"Jhon",
        "city": "Melbourne",
        "phoneNo":"1234567890"
    },
    {
        "personId":124,
        "name":"Amelia",
        "city": "Sydney",
        "phoneNo":"1234567890"
    },
    {
        "personId":125,
        "name":"Emily",
        "city": "Perth",
        "phoneNo":"1234567890"
    },
    {
        "personId":126,
        "name":"Abraham",
        "city": "Perth",
        "phoneNo":"1234567890"
    }
];
```

@autor: Victor Manuel Resendiz Cortina

Generar una tabla con javascript en un documento HTML que presente los datos.

Ver código en ejercicio7.js en el repositorio de GitHub

8.- Teniendo el siguiente código en un documento HTML:

```
<ul>
    <li id="C1" data-id="US" data-dial-code="1">USA</li>
    <li id="C2" data-id="CA" data-dial-code="1">Canada</li>
    <li id="C3" data-id="FF" data-dial-code="3">France</li>
</ul>
```

Genera la función javascript para que al dar click en cada elemento 'li' se muestre un alert con la siguiente información perteneciente al elemento

Elemento Seleccionado:
"ID elemento: {aquí el id}"
"ISO ID: {aquí el data-id}"
"Dial Code: {aquí el data-dial-code}"

Ver código en ejercicio8.js en el repositorio de GitHub

9.-Generar una pagina html y el código javascript para que:

a. Al mover el ratón en cualquier punto de la ventana del navegador, se muestre la posición del puntero respecto del navegador y respecto de la página:

Nota: utiliza las posiciones x, y del evento para los valores de la pagina

b. Al pulsar cualquier tecla, el mensaje mostrado debe cambiar para indicar el nuevo evento y su información asociada:

Ver código en ejercicio9.js en el repositorio de GitHub

10. A la pagina HTML de calculadora vista en CSS. Agrega las funcionalidades con javascript y manejo de eventos para: suma, resta, multiplicación, división, reset , resultado.

Ver código en ejercicio10.js en el repositorio de GitHub