Modificación de datos en arreglos

- Las **modificaciones** se pueden hacer **directamente** en los arreglos que pasamos como **parámetro** (solo funciona para arreglos y matrices, no para otros tipos de datos)

Explicación:

Ejemplo con variable.

1-Creamos una variable “global” y mostramos su valor
2- La modificamos dentro de una función y volvemos a mostrar su valor.
3- Por último mostramos el valor de la misma luego la función.

```
let variableNumerica:number=0;

console.log("valor de variableNumerica antes de la funcion: " +
variableNumerica);
modificarVariable(variableNumerica);

function modificarVariable(variableNumerica: number) {
    variableNumerica=1;
    console.log("valor de variableNumerica dentro de la funcion: "
+ variableNumerica);
}

console.log("valor de variableNumerica despues de la funcion: " +
variableNumerica);
```

Resultado:

```
valor de variableNumerica antes de la funcion: 0
valor de variableNumerica dentro de la funcion: 1
valor de variableNumerica despues de la funcion: 0
```

Como podemos observar, el valor de la variable global no se modifica.
No importa que tenga el mismo nombre dentro de la función.
variableNumerica dentro de la función (pertenece al scope local) por lo tanto TS la trata como otra variable.

Cuando se trata de arreglos no es lo mismo, veamos 2 ejemplos:

Ejemplo con arreglo.

1-Creamos un arreglo “global” y mostramos su valor
2- Lo pasamos por parámetro
3- En la función lo recibimos bajo el parámetro arregloParam al cual le asiganamos nuevos valores con la forma =[a,b,c] `arregloParam=[9,9,9];`

4-Mostramos los valores de arreglo y de arregloParam dentro de la función

5- Mostramos el valor de arreglo luego de la función

Cuando se trata de arreglos no es lo mismo, veamos 2 ejemplos:

```
let arreglo:number[]=[0,1,2];

console.log("valor de arreglo antes de la funcion: " + arreglo);
modificarVariable(arreglo);

function modificarVariable(arregloParam: number[]) {
    arregloParam=[9,9,9];//la reasignacion de valores crea un arreglo
nuevo en el ambito de la funcion
    //por lo tanto no modifica al arreglo "global"
    //let arreglo:number[]=[9,9,9];
    console.log("valor de arreglo dentro de la funcion: " + arreglo);
    console.log("valor de arregloParam: " + arregloParam);
}

console.log("valor de arreglo despues de la funcion: " + arreglo);
```

Resultado:

```
valor de arreglo antes de la funcion: 0,1,2
valor de arreglo dentro de la funcion: 0,1,2
valor de arregloParam: 9,9,9
valor de arreglo despues de la funcion: 0,1,2
```

Como podemos observar, el valor del arreglo no se modifica en ningún momento.

¿Pero qué es lo que sucede con arregloParam?

Bueno, la explicación más profunda la veremos adelante. Sin embargo nos sirve saber que la asignación del arreglo crea un nuevo arreglo LOCAL. Es decir no afecta al arreglo “global”.

Es como si estuviéramos haciendo `let arreglo:number[]=[9,9,9];`
Esta asignación no tiene efecto sobre el arreglo “global”.

Sigamos un poco más…

Cuando se trata de arreglos no es lo mismo, último ejemplo

Ejemplo con arreglo.

1-Creamos un arreglo “global” y mostramos su valor
2- Lo pasamos por parámetro
3- En la función lo recibimos bajo el parámetro **arregloParam** al cual **MODIFICAMOS** usando uno de los tantos métodos que podemos usar con arreglos.

4-Mostramos los valores de arreglo y de arregloParam dentro de la función

5- Mostramos el valor de arreglo luego de la función

```
let arreglo:number[]=[0,1,2];

console.log("valor de arreglo antes de la funcion: " + arreglo);

modificarVariable(arreglo);
function modificarVariable(arregloParam: number[]) {
    arregloParam.pop(); //el metodo pop elimina el ultimo elemento
del arreglo
    console.log("valor de arreglo dentro de la funcion: " + arreglo);
    console.log("valor de arregloParam: " + arregloParam);
}

console.log("valor de arreglo despues de la funcion: " + arreglo);//
NO mantiene el valor original
```

Al ejecutar este código obtenemos lo siguiente:

```
valor de arreglo antes de la funcion: 0,1,2
valor de arreglo dentro de la funcion: 0,1
valor de arregloParam: 0,1
valor de arreglo despues de la funcion: 0,1
```

Como podemos ver ahora el arreglo “global” si se ve afectado.

¿Qué está pasando?

Bueno, cuando pasamos por parámetro un arreglo no estamos pasando una copia del mismo, sino una referencia a él (esto también lo veremos en detalle más adelante).

Por el momento quedémonos con que pasamos el arreglo mismo.
Dentro de la declaración de la función:

```
function modificarVariable(arregloParam: number[]) {
```

Vemos que recibimos al arreglo como parámetro con el nombre arregloParam (esto es a propósito para demostrar que no hace falta que tenga el mismo nombre),
para TS se trata del mismísimo arreglo original, el “global”.(esto también es así para el ejemplo anterior).
La diferencia está en la MODIFICACION del arregloParam vs la reasignación.
La modificación que hagamos dentro de una función (o bloque) afectarán al arreglo original.