#### ****1. length****: ****El atributo length almacena el número de elementos del vector****

#### ****2. push()****: ****El método push() añade uno o más elementos al final del vector****

#### Ejemplo de vector Push

function aniadirNumero() {

console.log("Vector Original (Vector1):" ,vector1);

vector1.push(10, 13);

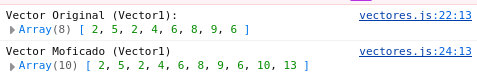
console.log("Vector Moficado (Vector1)",vector1);

}

const bntNumero = document.getElementById("btnaniadirN");

bntNumero.addEventListener("click", aniadirNumero);

Resultado:



#### ****3. pop() :**** ****elimina el último elemento del array y lo puede devolver****

function eliminarN(){

console.log("Vector1 original" ,vector1);

vector1.pop()

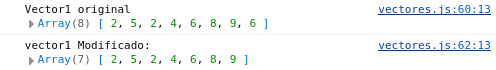
console.log("vector1 Modificado:" ,vector1);

}

const btnEliminarN=document.getElementById("btnEliminarN");

btnEliminarN.addEventListener("click",eliminarN);

Resultado:



#### ****4. unshift(item1,item2...)****: a****ñade uno o más elementos al comienzo del array****

function aniadirPrincipio(){

console.log("Original" ,vector1);

vector1.unshift(1,3);

console.log("Modificado",vector1);

}

const btnAniadirU=document.getElementById("btnAniadirU");

btnAniadirU.addEventListener("click",aniadirPrincipio);

Resultado:



#### ****5. shift() : elimina el primer elemento del array y lo retorna****

function eliminarPrincipio(){

console.log("Vector 2 Original" ,vector2);

const primerElemento=vector2.shift();

console.log("Elemento borrado de vector2: ",primerElemento);

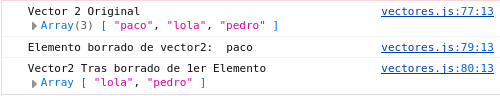
console.log("Vector2 Tras borrado de 1er Elemento",vector2)

}

const btnEliminarS=document.getElementById("btnEliminarS");

btnEliminarS.addEventListener("click",eliminarPrincipio);

Resultado:



#### 6 ****concat(item1,item2...) retorna un nuevo array que es la concatenación de los**** [**arrays**](https://moodle.educarex.es/evex/mod/page/view.php?id=132980) ****que se pasan como parámetros.****

#### const vector1 = [1,2,3]

#### const vector2 = [4,5,6]

#### const vector3 = [7,8,9]

#### const vectorTotal = vector1.concat(vector2, vector3)

#### idéntico a lo anterior pero con la sintaxis spread (expandir)

#### const vector1 = [1, 2, 3]

#### const vector2 = [4, 5, 6]

#### const vectorTotal = [...vector1, ...vector2]

#### 7. slice ( inicio, fin) : devuelve una subcadena desde la posición de inicio hasta fin (no se incluye y éste es opcional), Si solo aparece inicio se entiende que es hasta el final.

function copiarVector(){

console.log("Original",vector1)

const nuevoVector1=vector1.slice(2,6)

console.log("Nuevo Vector1",nuevoVector1)

}

const btnSlice=document.getElementById("btnSlice");

btnSlice.addEventListener("click",copiarVector);

Resultado:



#### si no se pasan parámetros se hace una copia exacta

function copiarVector(){

console.log("Original",vector1)

const copiavector1=vector1.slice()

console.log("copia Vector1",copiavector1)

}

const btnSlice=document.getElementById("btnSlice");

btnSlice.addEventListener("click",copiarVector);

Resultado:



#### ****8.splice(inicio, cantidadBorrar...): elimina y añade elementos a un vector.****

#### const vector = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

#### vector.splice(3, 4)

#### también puede retornar un array con los elementos borrados:

#### vector2=vector.splice(2,3) También añade elementos a partir de una posición x. En este ejemplo se eliminan los dos primeros elementos y se añaden a continuación tres elementos

const vector = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

vector.splice(0, 2, 5, 5, 5)

#### También se puede añadir sin necesidad de borrar

const vector = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

arreglo.splice(5, 0, 1, 1, 1)

**9.fill(valor,comienzo,fin) El método fill rellena el array con el valor que se pasa como parámetro,**

**sustituyendo los valores actuales del vector.**

const vector = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

vector.fill(10)

se puede especificar el valor y desde qué elemento (lo hará hasta el final)

vector.fill(20,5)

también se puede especificar un tercer parámetro indicando la cantidad de elementos

a sustituir

vector.fill(20,5,2)

**10. join (separador) método join retorna un string con todos**

**los elementos del vector separados por el separador que se pasa como parámetro.**

Podemos pasar como separador un ";" y luego nos retorna un string con el contenido de cada componente

separado por dicho separador:

function separadorJoin(){

console.log("Vector2 Original",vector2);

const cadena=vector2.join(';')

console.log("Vector 2 con join Aplicado: ",cadena)

}

const btnJoin=document.getElementById("btnJoin");

btnJoin.addEventListener("click",separadorJoin);

Resultado:



#### ****11. Reverse() :invierte el orden de un vector****. Modifica el original

function ordenarInverso(){

console.log("Vector1 Original: " ,vector1);

const vectorCopia=vector1.slice().sort().reverse();

console.log("Vector1 Copia con Revese",vectorCopia)

}

const btnReverse=document.getElementById("btnReverse");

btnReverse.addEventListener("click",ordenarInverso);

Resultado:



#### 12 sort() :ordena el contenido del vector

function ordenar(){

console.log("Vector1 Original: " ,vector1);

const vectorCopia=vector1.slice().sort((a, b)=> a -b );

console.log("Vector1 Copia con Sort",vectorCopia)

}

const btnSort=document.getElementById("btnSort");

btnSort.addEventListener("click",ordenar);

Resultado:



#### ****13. vector.includes() : retorna true si el elemento se encuentra dentro de una lista****

function encontrar(){

console.log("Vector 2 Original:", vector2);

if (vector2.includes("pedro")) {

console.log("El nombre 'pedro' está en la lista:", "pedro");

} else {

console.log("El nombre 'pedro' no existe en la lista");

}

}

const btnIncludes=document.getElementById("btnIncludes");

btnIncludes.addEventListener("click",encontrar);

Resultado:



#### ****14.vector.indexof ( ) retorna un entero que representa el índice del elemento en el array. Si no se encuentra el elemento, retorna -1****

#### const meses = ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo'] console.log(meses.indexOf("Junio"))

#### 15 vector.forEach() : ejecuta la función que le pasamos como parámetro. La función recibe el elemento, el índice (opcional) y el vector (opcional).

#### const alumnos = [{ nombre: 'Paco', edad: 18 }, { nombre: 'Mou', edad: 20 }, { nombre: 'Andrés', edad: 19 }]

#### let alMasEdad = alumnos[0]

#### alumnos.forEach(item => {

#### if (item.edad > alMasEdad.edad)

#### { alMasEdad = item }

#### })

#### ****16. vector.map(funcion) : retorna un nuevo array con los resultados de aplicar la función que se pasa como parámetro a cada elemento del array. El vector original no se modifica. Se consigue un nuevo vector con las modificaciones****

function mapeo(){

console.log("Vector 3 Original",vector3);

const vector3min = vector3.map(peli =>

{

if (peli.minutos < 100)

{ return peli;

}

//establecemos que undifined es vacio ya que descartamos el resto

//del array

else{

return undefined={};

}

})

console.log("Resultado de Map: " ,vector3min);

}

const btnMap=document.getElementById("btnMap");

btnMap.addEventListener("click",mapeo);

Resultado:



#### ****17. vector.filter(funcion) devuelve un nuevo array con los elementos que cumplen con la condición de la función que se pasa como parámetro.****

function filtrar(){

console.log("Vector 3 Original",vector3);

const vector3min=vector3.filter(min=>min.minutos > 100);

console.log("Resultado Vector3 Filter",vector3min);

}

const btnFilter=document.getElementById("btnFilter");

btnFilter.addEventListener("click",filtrar);

Resultado:



#### 18. vector.reduce() reduce el array a un único valor aplicando la función que se pasa como parámetro (el método reduce no modifica el contenido del vector)

function reducir(){

console.log("Vector 1 Original" ,vector1);

const totalVector=vector1.reduce((suma,item)=>suma+item,0);

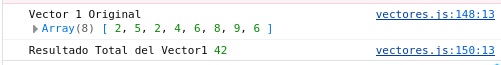
console.log("Resultado Total del Vector1",totalVector)

}

const btnReduce=document.getElementById("btnReduce");

btnReduce.addEventListener("click",reducir);

Resultado:



***Más Funciones de Array***

### some()

**Descripción**: Verifica si **al menos un elemento** del array cumple con una condición. Devuelve true o false.

function par() {

console.log("Vector 1 Original:", vector1);

const numeroPar = vector1.some(num => num % 2 === 0);

if (numeroPar) {

//usamos filter para filtrar los pares

const numerosPares = vector1.filter(num => num % 2 === 0);

console.log("Sí hay números pares",numerosPares);

} else {

console.log("No hay números pares");

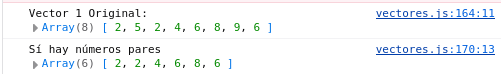
}

}

const btnSome = document.getElementById("btnSome");

btnSome.addEventListener("click", par);

Resultado:



### every()

**Descripción**: Verifica si **todos los elementos** del array cumplen una condición. También devuelve true o false.

function mayormenorEdad() {

console.log("Vector 4 Original:", vector4);

if (vector4.every(edad => edad >= 18)) {

console.log("Todos son mayores de edad:", vector4);

} else {

console.log("No todos son mayores de edad:", vector4.filter(edad=>edad.edad=='17'));

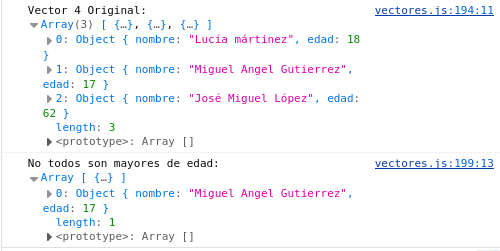
}

}

const btnEvery = document.getElementById("btnEvery");

btnEvery.addEventListener("click", mayormenorEdad);

Resultado:



### find()

**Descripción**: Retorna el **primer elemento** que cumpla con una condición. Si no encuentra ninguno, retorna undefined.

function buscar() {

console.log("Vector 3 Original:", vector3);

const pelimenorduracion = vector3.find(min => min.minutos < 100);

if (pelimenorduracion) {

console.log("La película que dura menos es:",

pelimenorduracion.titulo,

"con duración de:",

pelimenorduracion.minutos,

"minutos"

);

} else {

console.log("No existe la película con esas características");

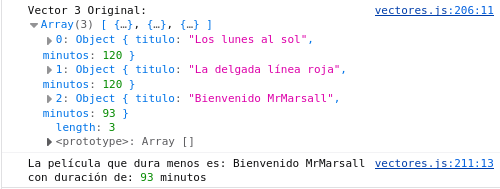
}

}

const btnFind = document.getElementById("btnFind");

btnFind.addEventListener("click", buscar);

Resultado:



### Object.keys()

**Descripción**: Retorna un array con las **claves** (keys) de un objeto.

function identificadores() {

console.log("Vector 4 Original:", vector4);

//ponemos el id del objeto a identificar si no nos devolvera

//Arrays > [{0 ,1 ,2}] en lugar de Arrays > [{nombre , edad }] ya que

// quiero mostrar estos dos ultimos nombre y edad

const claves = Object.keys(vector4[1]);

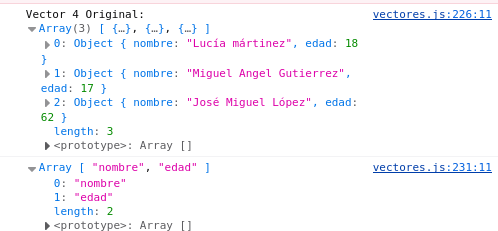
console.log(claves)

}

const btnKeys = document.getElementById("btnKeys");

btnKeys.addEventListener("click", identificadores);

Resultado:



### Object.entries() Descripción:Devuelve un array de arrays, donde cada subarray contiene un par [clave, valor].

function entrada(){

console.log("Vector 3 Original",vector3);

console.log("Vector 3 con entradas de objetos:",Object.entries(vector3));

}

const btnEntry = document.getElementById("btnEntry");

btnEntry.addEventListener("click", entrada);

Resultado:

