

Vectores (Arrays) en JavaScript

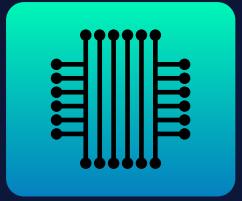


Tabla de Contenido

- 01** Definicion de que es vector
- 02** Acceso y Modificación de Elementos
- 03** Métodos Básicos de Vectores
- 04** Iteración a través de Vectores
- 05** Métodos Avanzados de Vectores
- 06** Juego
- 07** Reto
- 08** Comida

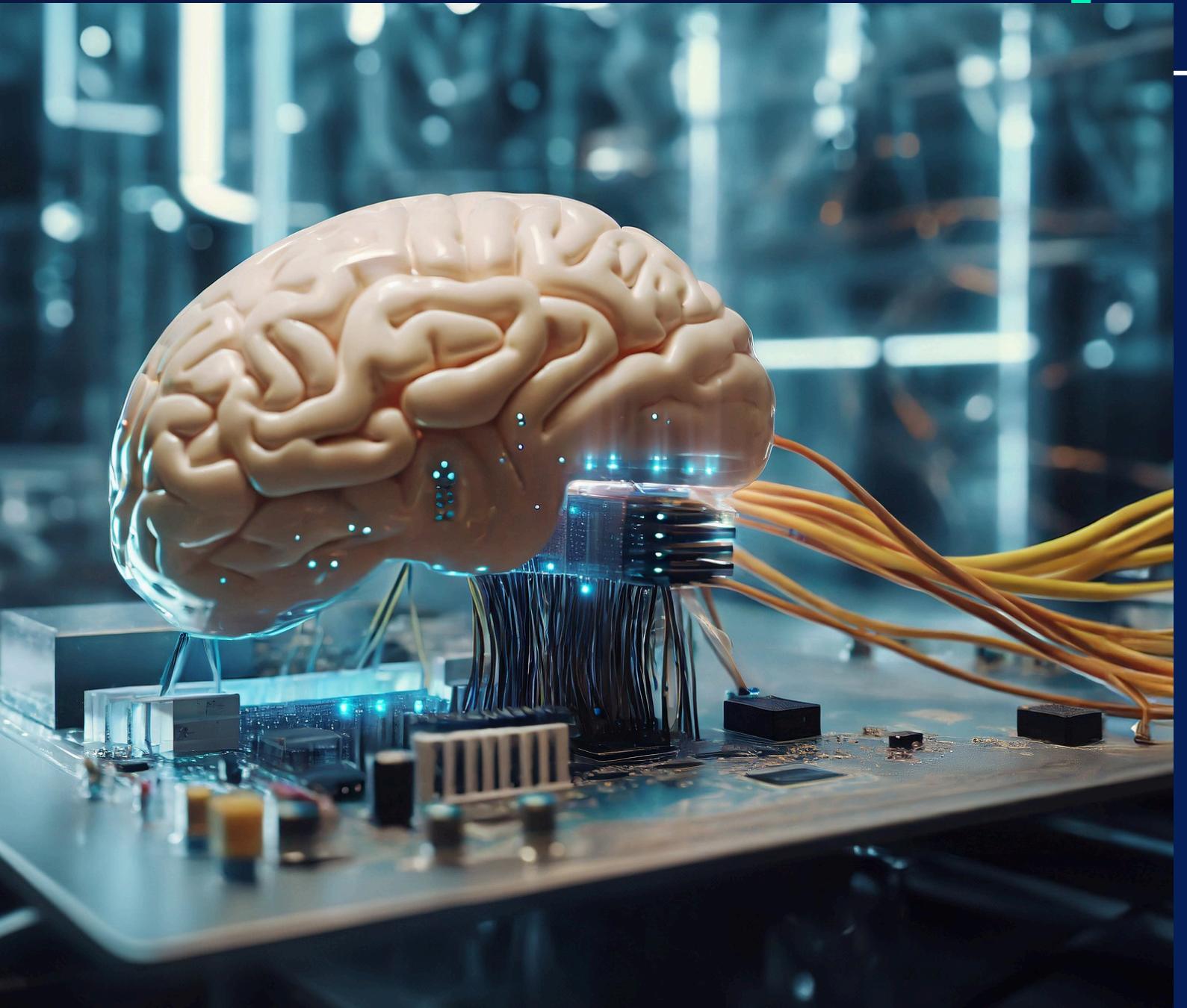


Que es un vector



Definición

Un vector (array) en JavaScript es una estructura de datos que almacena una colección de elementos de cualquier tipo, accesibles por índices numéricos que comienzan en 0. Se usa para gestionar listas de valores relacionados.





Ejemplo

```
// Declaración de un vector con elementos
let frutas = ['manzana', 'banana', 'naranja'];

// Acceso a elementos del vector
console.log(frutas[0]); // Imprime "manzana"
console.log(frutas[1]); // Imprime "banana"
console.log(frutas[2]); // Imprime "naranja"

// Modificación de un elemento del vector
frutas[1] = 'pera';
console.log(frutas); // Imprime ["manzana",
"pera", "naranja"]

// Añadir un nuevo elemento al final del vector
frutas.push('uva');
console.log(frutas); // Imprime ["manzana",
"pera", "naranja", "uva"]

// Eliminar el último elemento del vector
frutas.pop();
console.log(frutas); // Imprime ["manzana",
"pera", "naranja"]
```



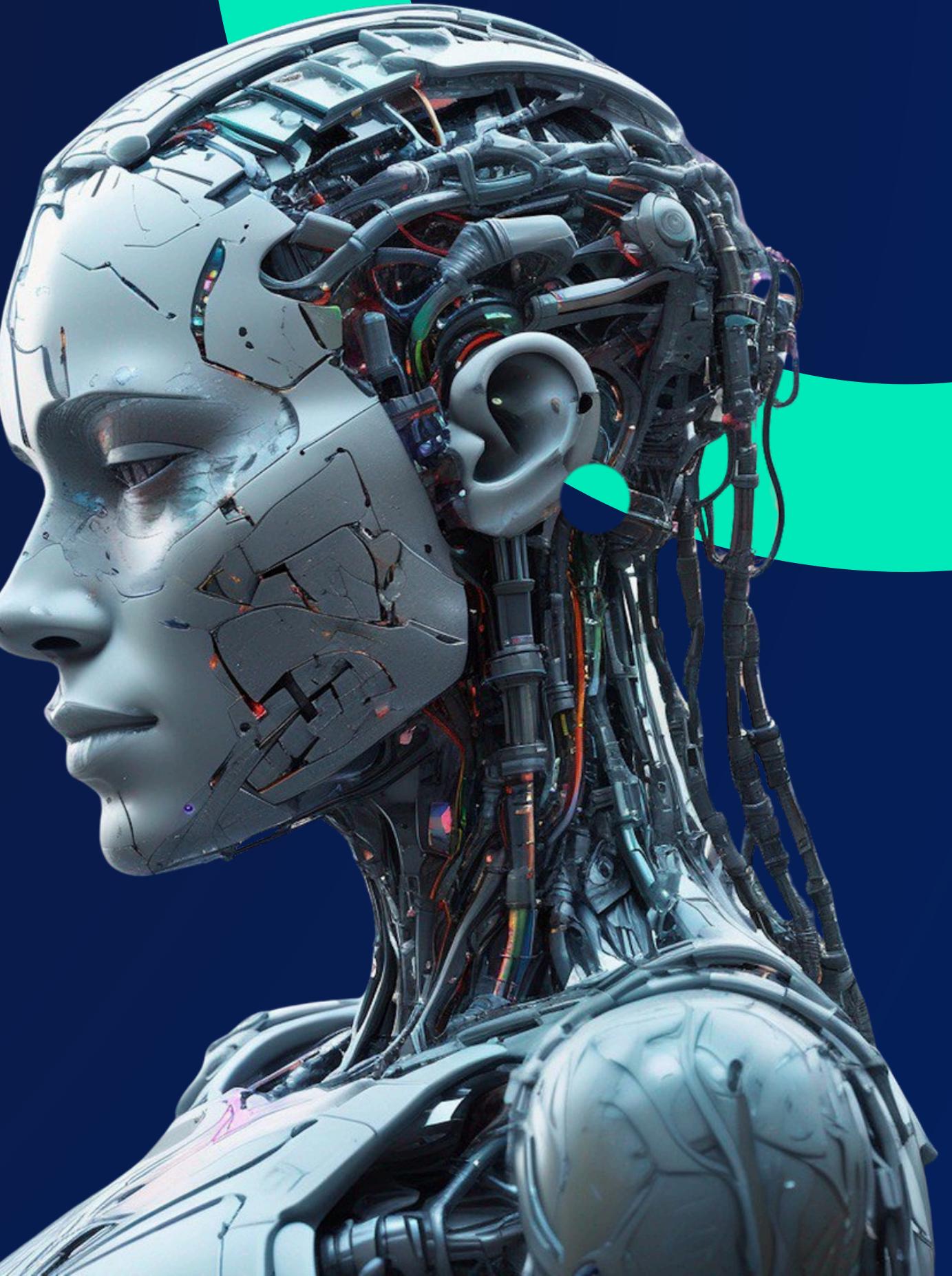
Acceso y Modificación de Elementos

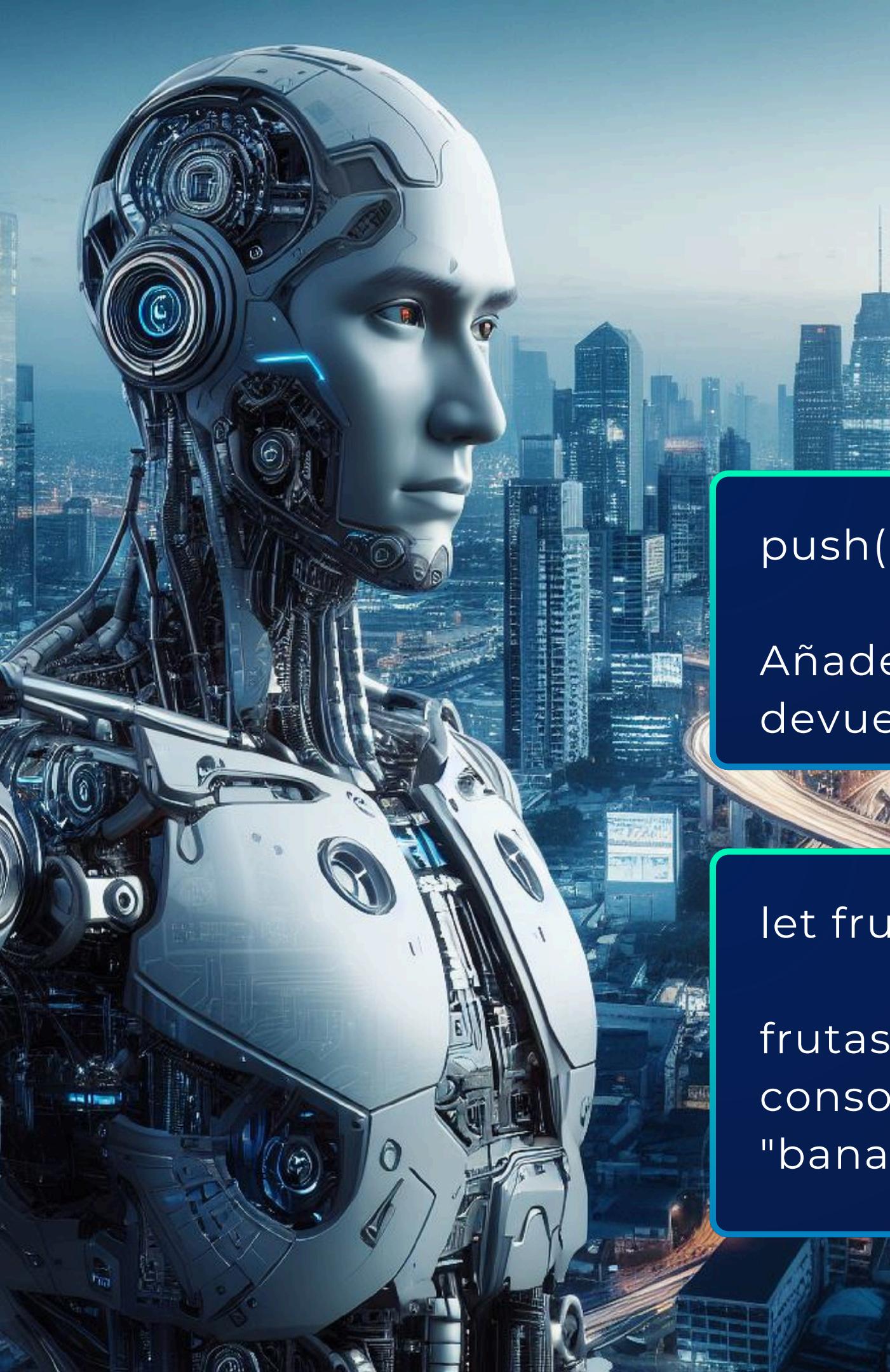
Puedes acceder a los elementos de un array utilizando su índice. Los índices comienzan en 0, por lo que el primer elemento tiene el índice 0, el segundo elemento el índice 1, y así sucesivamente.

```
// Declarar un array
let frutas = ['manzana', 'banana', 'naranja'];
// Acceder a los elementos del array
console.log(frutas[0]); // Imprime "manzana"
console.log(frutas[1]); // Imprime "banana"
console.log(frutas[2]); // Imprime "naranja"
```

```
// Declarar un array
let frutas = ['manzana', 'banana', 'naranja'];
// Modificar un elemento del array
frutas[1] = 'pera';
// Imprimir el array modificado
console.log(frutas); // Imprime ["manzana", "pera",
"naranja"]
```

× × ×





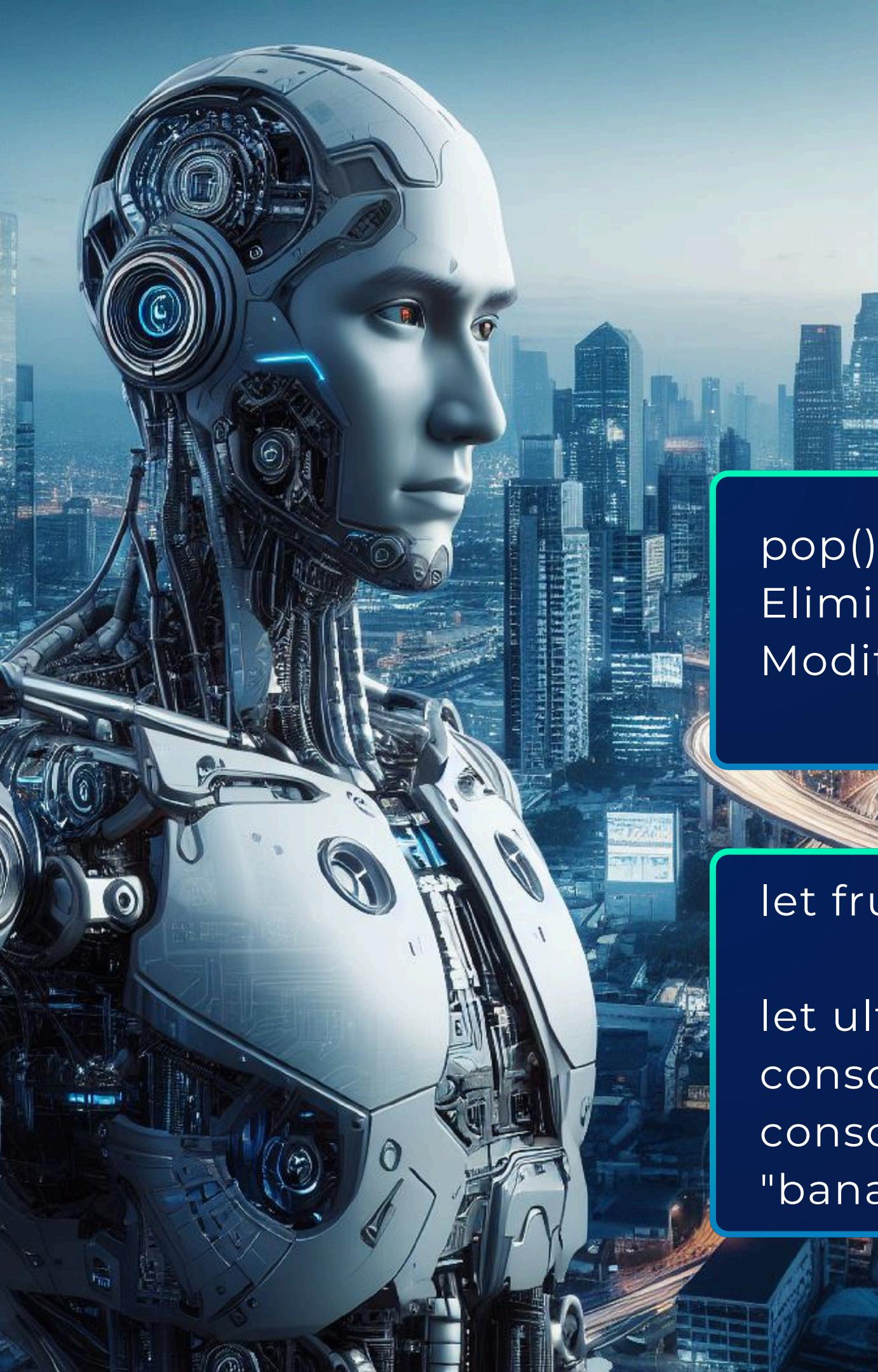
Métodos Básicos de Vectores

`push():`

Añade uno o más elementos al final del array y devuelve la nueva longitud del array.

```
let frutas = ['manzana', 'banana'];
```

```
frutas.push('naranja', 'uva');  
console.log(frutas); // Imprime ["manzana",  
"banana", "naranja", "uva"]
```



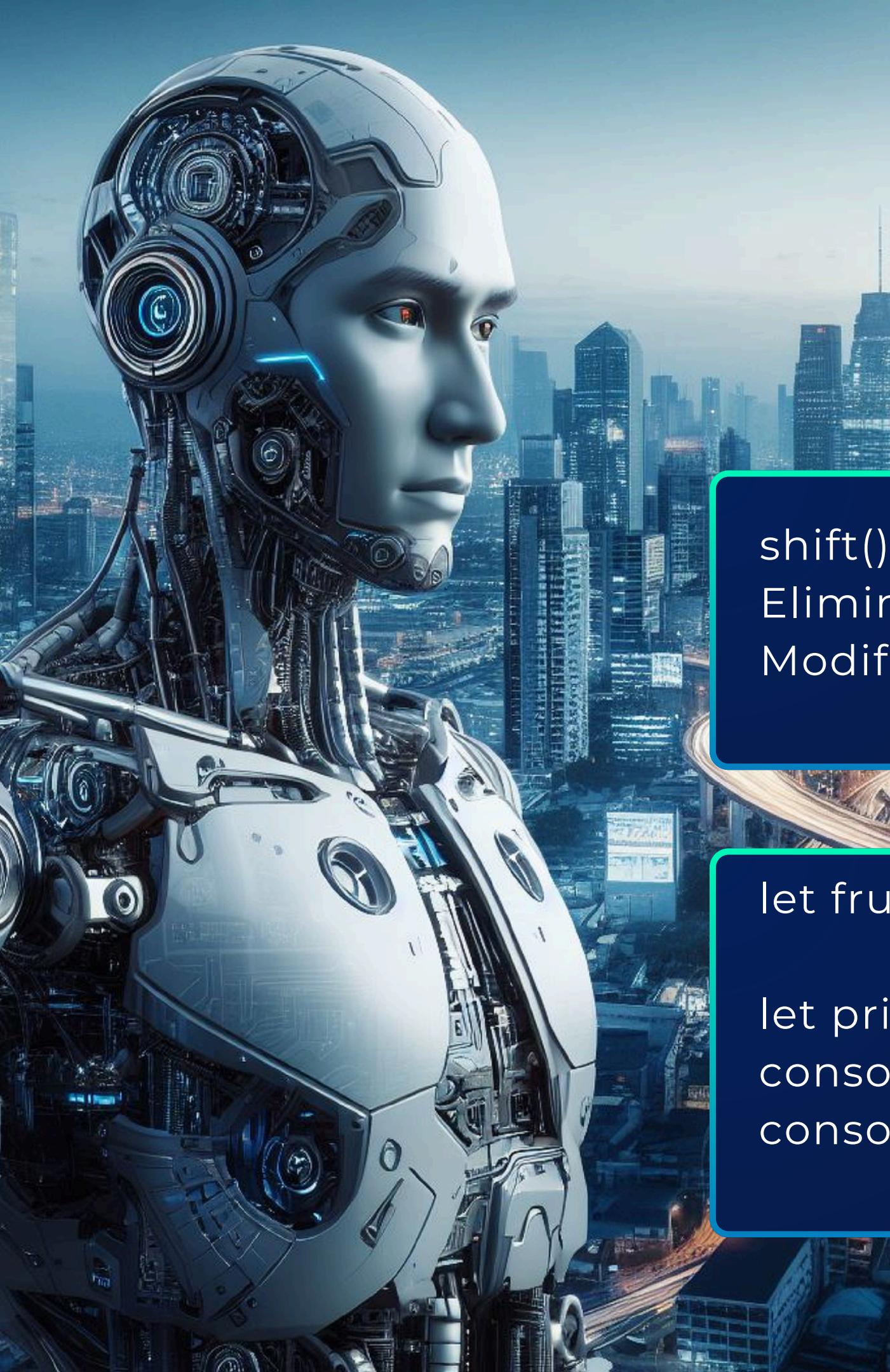
Métodos Básicos de Vectores

pop()

Elimina el último elemento del array y lo devuelve.
Modifica la longitud del array.

```
let frutas = ['manzana', 'banana', 'naranja'];
```

```
let ultimaFruta = frutas.pop();
console.log(ultimaFruta); // Imprime "naranja"
console.log(frutas); // Imprime ["manzana",
"banana"]
```



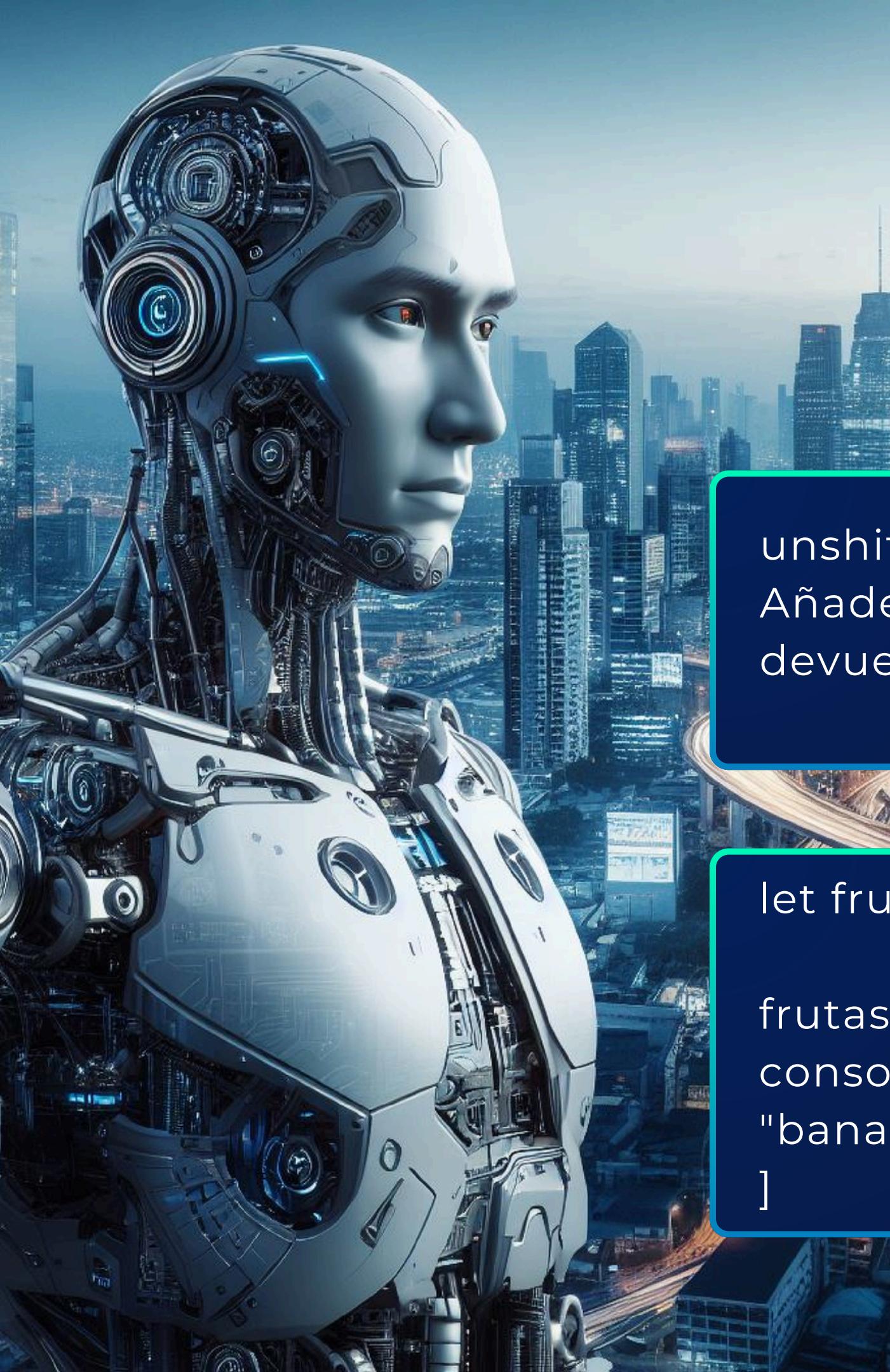
Métodos Básicos de Vectores

`shift()`

Elimina el primer elemento del array y lo devuelve.
Modifica la longitud del array.

```
let frutas = ['manzana', 'banana', 'naranja'];
```

```
let primeraFruta = frutas.shift();
console.log(primeraFruta); // Imprime "manzana"
console.log(frutas); // Imprime ["banana", "naranja"]
```



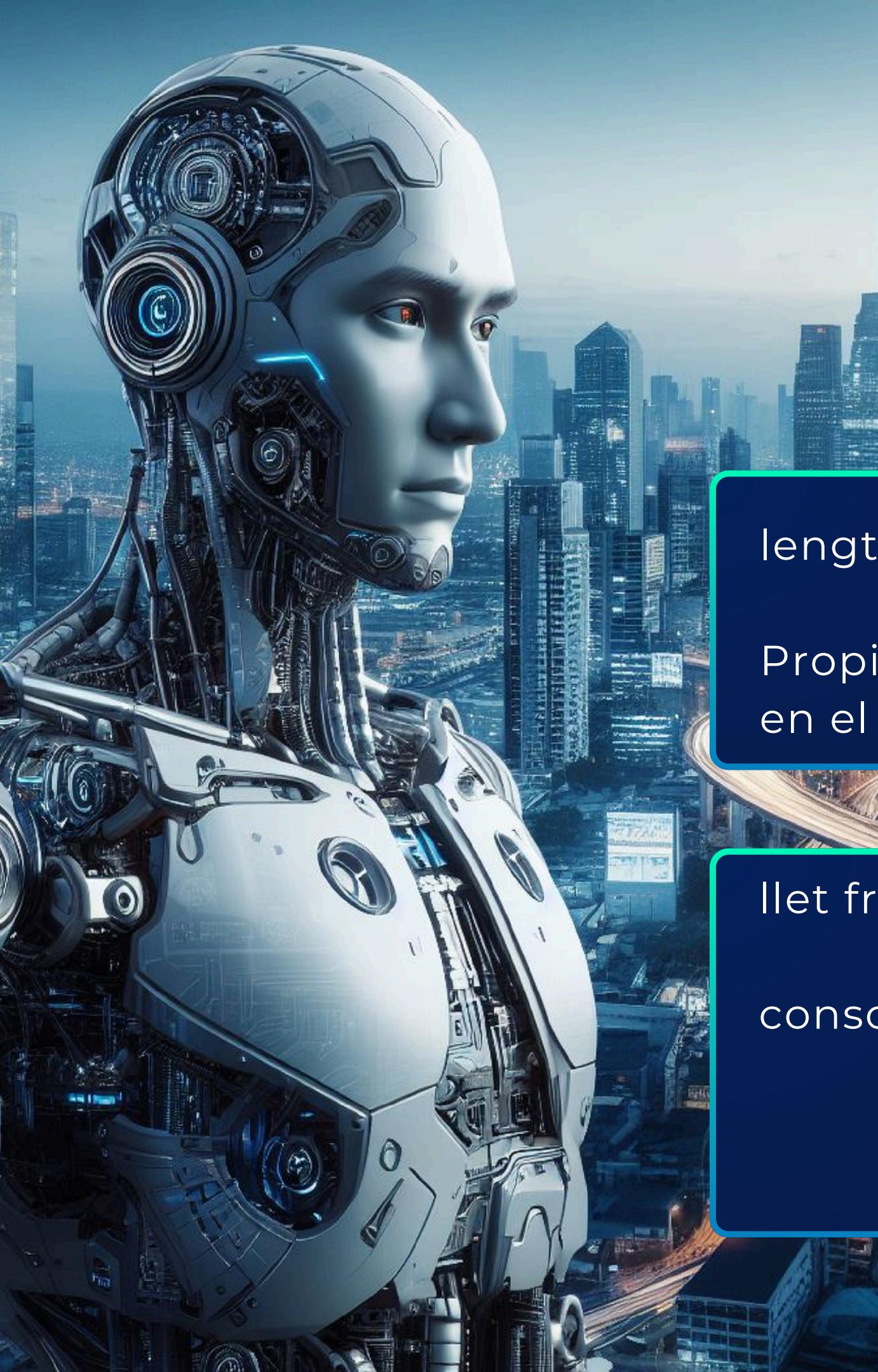
Métodos Básicos de Vectores

`unshift()`

Añade uno o más elementos al inicio del array y devuelve la nueva longitud del array.

```
let frutas = ['banana', 'naranja'];
```

```
frutas.unshift('manzana');
console.log(frutas); // Imprime ["manzana",
"banana", "naranja"]
]
```



Métodos Básicos de Vectores

length

Propiedad que devuelve la cantidad de elementos en el array.

```
let frutas = ['manzana', 'banana', 'naranja'];
```

```
console.log(frutas.length); // Imprime 3
```

Iteración a través de Vectores

Bucle for

El bucle for es una de las formas más básicas y flexibles para iterar sobre los elementos de un array

```
let frutas = ['manzana', 'banana', 'naranja'];
```

```
for (let i = 0; i < frutas.length; i++) {  
    console.log(frutas[i]);  
}
```

```
// Imprime:  
// manzana  
// banana  
// naranja
```

Iteración a través de Vectores

Bucle for...of

El bucle for...of es una forma más moderna y concisa para iterar sobre los elementos de un array.

```
let frutas = ['manzana', 'banana', 'naranja'];

for (let fruta of frutas) {
    console.log(fruta);
}

// Imprime:
// manzana
// banana
// naranja
```

Iteración a través de Vectores

Método **forEach()**

El método `forEach()` ejecuta una función en cada elemento del array. Es una opción más funcional para iterar sobre arrays.

```
let frutas = ['manzana', 'banana', 'naranja'];

frutas.forEach(function(fruta) {
  console.log(fruta);
});

// Imprime:
// manzana
// banana
// naranja
```

Métodos Avanzados de Vectores

map()

El método map() crea un nuevo array con los resultados de aplicar una función a cada elemento del array original.

```
let numeros = [1, 2, 3, 4, 5];  
  
// Crear un nuevo array con cada número multiplicado por 2  
let dobles = numeros.map(function(numero) {  
    return numero * 2;  
});  
  
console.log(dobles); // Imprime [2, 4, 6, 8, 10]
```

Métodos Avanzados de Vectores

filter()

El método filter() crea un nuevo array con todos los elementos que cumplan una condición especificada en una función.

```
let numeros = [1, 2, 3, 4, 5];  
  
// Crear un nuevo array con los números  
mayores que 3  
let mayoresQueTres =  
numeros.filter(function(numero) {  
    return numero > 3;  
});  
  
console.log(mayoresQueTres); // Imprime [4, 5]
```

Métodos Avanzados de Vectores

reduce()

El método `reduce()` aplica una función a un acumulador y a cada elemento del array (de izquierda a derecha) para reducirlo a un solo valor.

```
let numeros = [1, 2, 3, 4, 5];  
  
// Calcular la suma de todos los números  
let suma =  
numeros.reduce(function(acumulador,  
numero) {  
    return acumulador + numero;  
}, 0);  
  
console.log(suma); // Imprime 15  
]
```

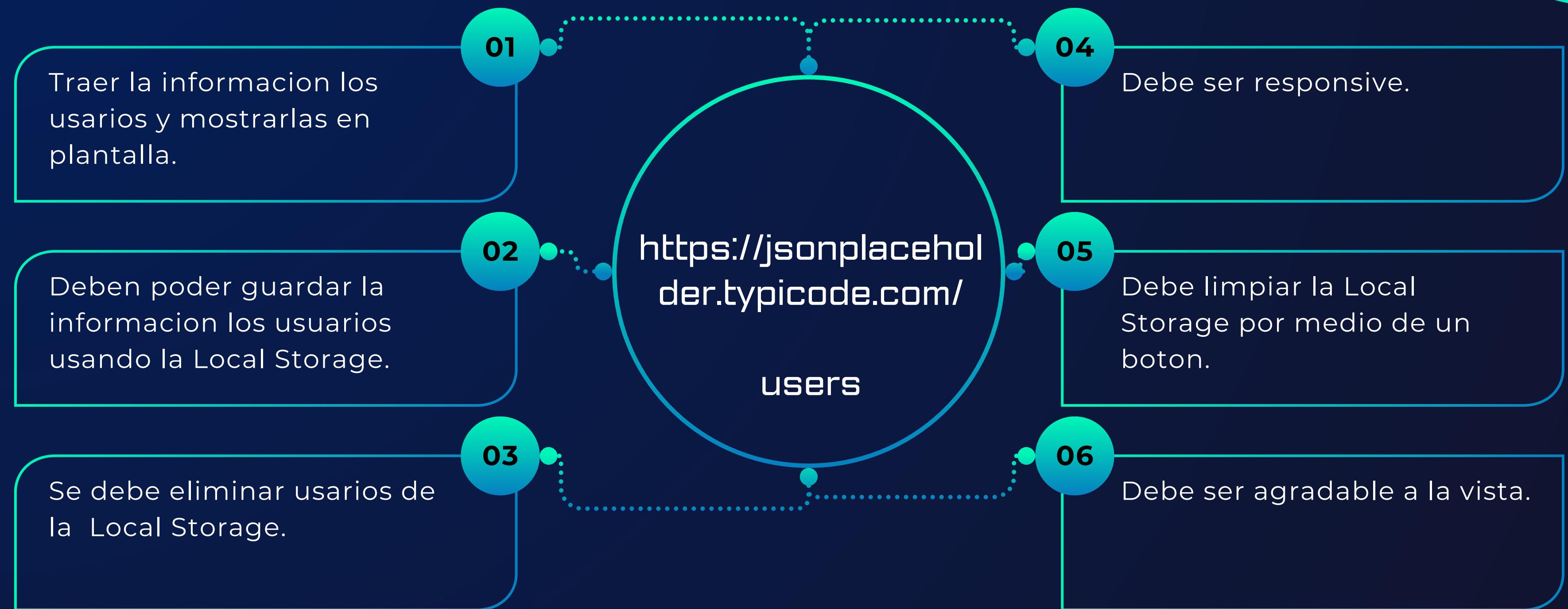
Métodos Avanzados de Vectores

find()

El método `find()` devuelve el primer elemento que cumple con la condición especificada en una función. Si ningún elemento cumple con la condición, devuelve `undefined`

```
let numeros = [1, 2, 3, 4, 5];  
  
// Encontrar el primer número mayor que 3  
let mayorQueTres =  
numeros.find(function(numero) {  
    return numero > 3;  
});  
  
console.log(mayorQueTres); // Imprime 4
```

Proyecto



60-30-10

Azul y Neutros

- 60%: Azul Marino (#2C3E50)
- 30%: Gris Claro (#BDC3C7)
- 10%: Naranja (#E67E22)

XXX

Morado y Pastel

- 60%: Morado (#8E44AD)
- 30%: Lila Claro (#D7BDE2)
- 10%: Amarillo (#F7DC6F)

Verde y Tierra

- 60%: Verde Oscuro (#27AE60)
- 30%: Beige (#F1C40F)
- 10%: Marrón (#8E44AD)

Rojo y Gris

- 60%: Rojo (#C0392B)
- 30%: Gris Medio (#7F8C8D)
- 10%: Blanco (#ECF0F1)

XXX



Muchas Gracias

Contáctanos si tienes alguna duda.

Página de Recursos

