

#MeGustaProgramar

# Arreglos o Matrices

Ing. Christian Arévalo Jesús

Tema 07 – Arreglos o Matrices



# ¿Qué es una matriz?

**Objetos**

**Multiples  
valores**



**Almacenar**

**Acceder  
individual**

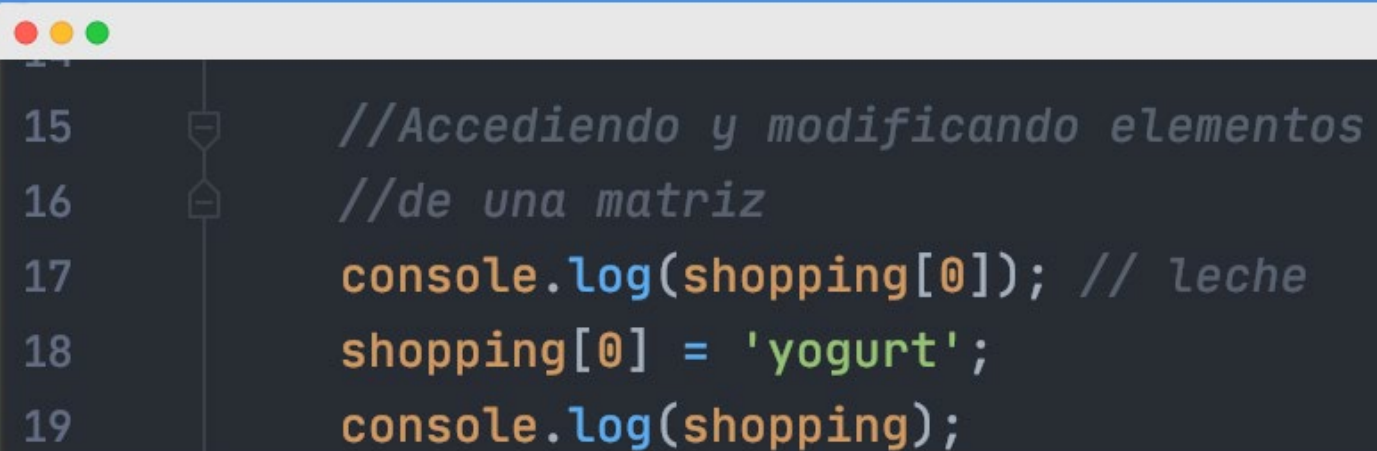
**Recorrerlo**

# Creando una matriz

```
10     let shopping = ['leche', 'pan', 'azucar', 'aceite'];  
11     console.log(shopping);  
12     let sequence = [1,2,3,4,5,6,7,8,9];  
13     let random = ['hola', true, 12, 21, ['a','b',1,2]];
```

# Accediendo y modificando un elemento

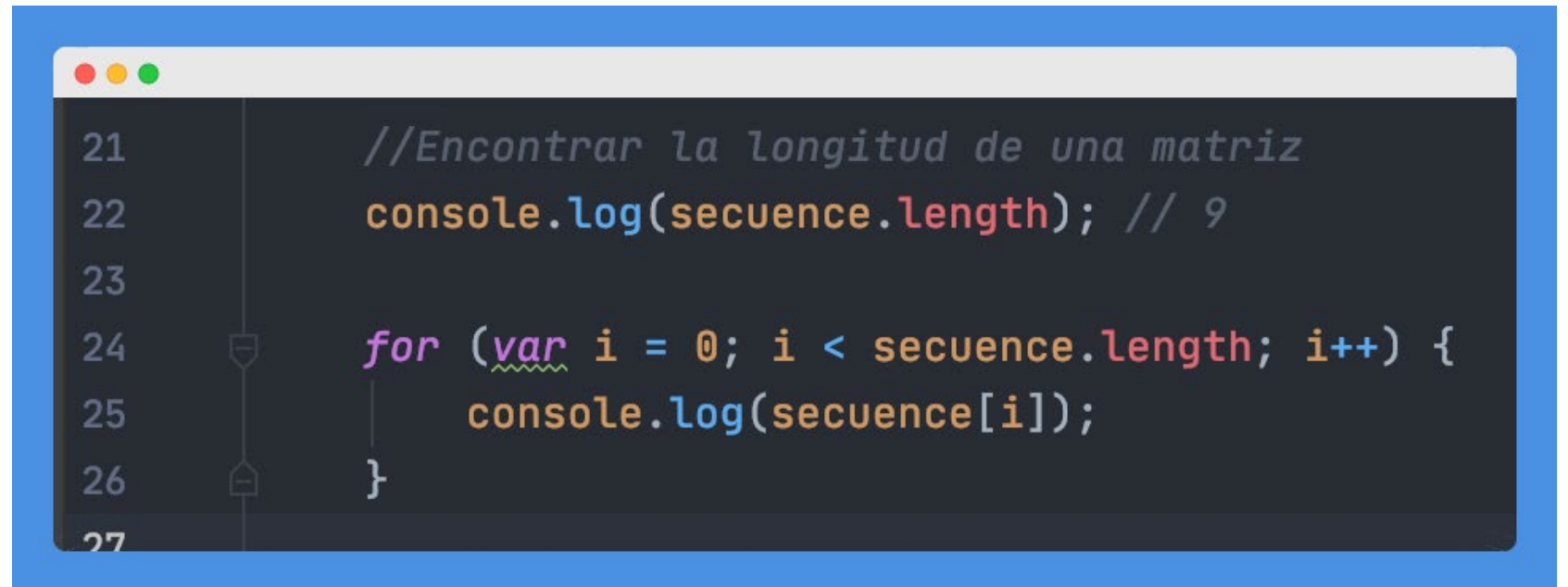
Puedes entonces acceder a elementos individuales en la matriz mediante la notación de corchetes



```
14  
15 //Accediendo y modificando elementos  
16 //de una matriz  
17 console.log(shopping[0]); // leche  
18 shopping[0] = 'yogurt';  
19 console.log(shopping);
```


# Encontrar la longitud de una matriz

Puedes averiguar la longitud de una matriz (cuántos elementos contiene) exactamente de la misma manera que determinas la longitud (en caracteres) de una cadena — utilizando la propiedad *length*.

A screenshot of a code editor window with a blue border. The editor has a dark background and shows JavaScript code. On the left, line numbers 21 through 27 are visible. The code consists of a comment on line 21, a log statement on line 22, and a for loop on lines 24-26. The variable 'sequence' is used throughout. The code is as follows:


```
21 //Encontrar la longitud de una matriz
22 console.log(sequence.length); // 9
23
24 for (var i = 0; i < sequence.length; i++) {
25     console.log(sequence[i]);
26 }
27
```

# Conversión entre matrices y cadenas

A code editor window with a blue border and a light gray title bar containing three colored window control buttons (red, yellow, green). The editor has a dark background with syntax-highlighted JavaScript code. Line numbers 28, 29, 30, and 31 are visible on the left side of the code area.

```
28      //conversion de matrices y cadenas
29      let myData = 'Perú,Brasil,Argentina,Colombia';
30      let country = myData.split(',');
31      console.log(country);
```

# Conversión entre matricez y cadenas



```
33 //de lo contrario
34 console.log(country.join(','));
35
36 let dogNames = ['Rocket','Flash','Bella','Slugger'];
37 dogNames.toString(); //Rocket,Flash,Bella,Slugger
38
```

# Agregar o eliminar elementos de la matriz

Antes que nada, para añadir o eliminar un elemento al final de una matriz podemos usar [push\(\)](#) y [pop\(\)](#) respectivamente

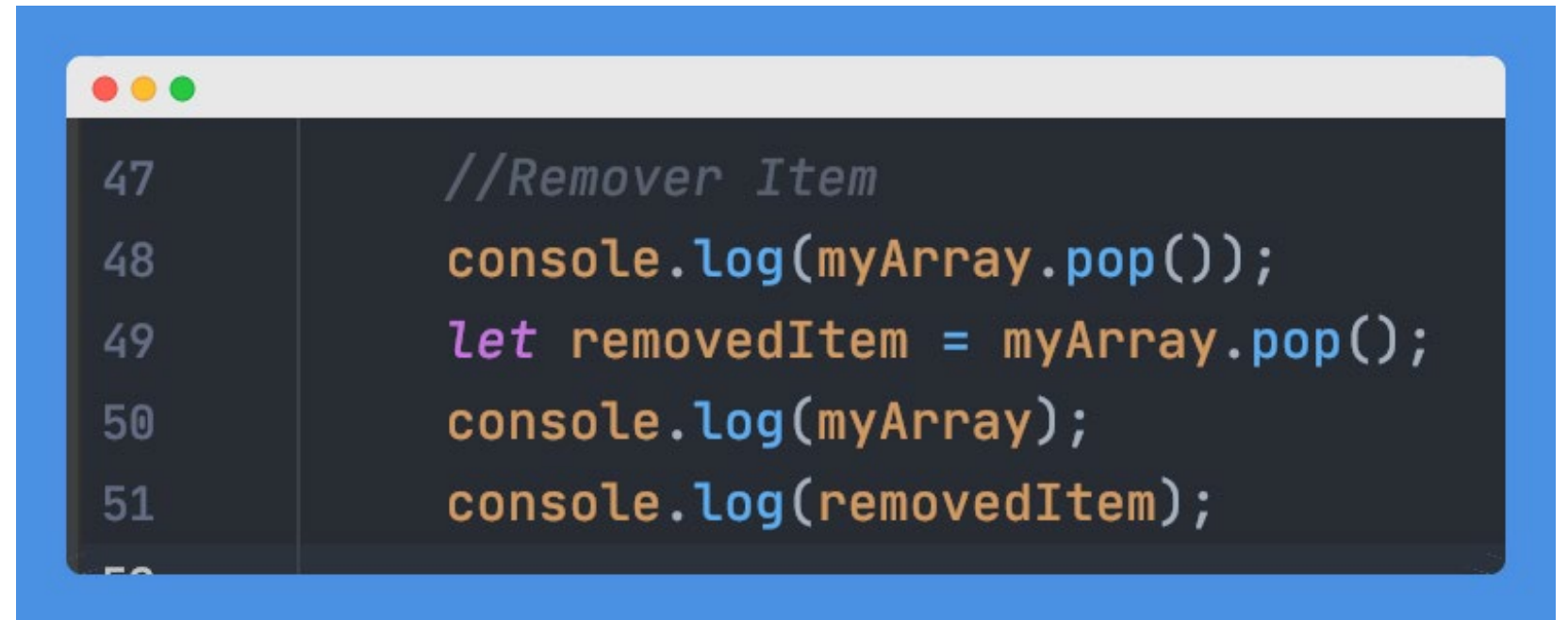


```
39 //Agregar elementos de una matriz
40 let myArray = ['Christian', 'Pedro', 'Juan', 'Andrea'];
41
42 myArray.push('Luis');
43 console.log(myArray);
44 myArray.push('Andres', 'Marcos');
45 console.log(myArray);
```



# Agregar o eliminar elementos de la matriz

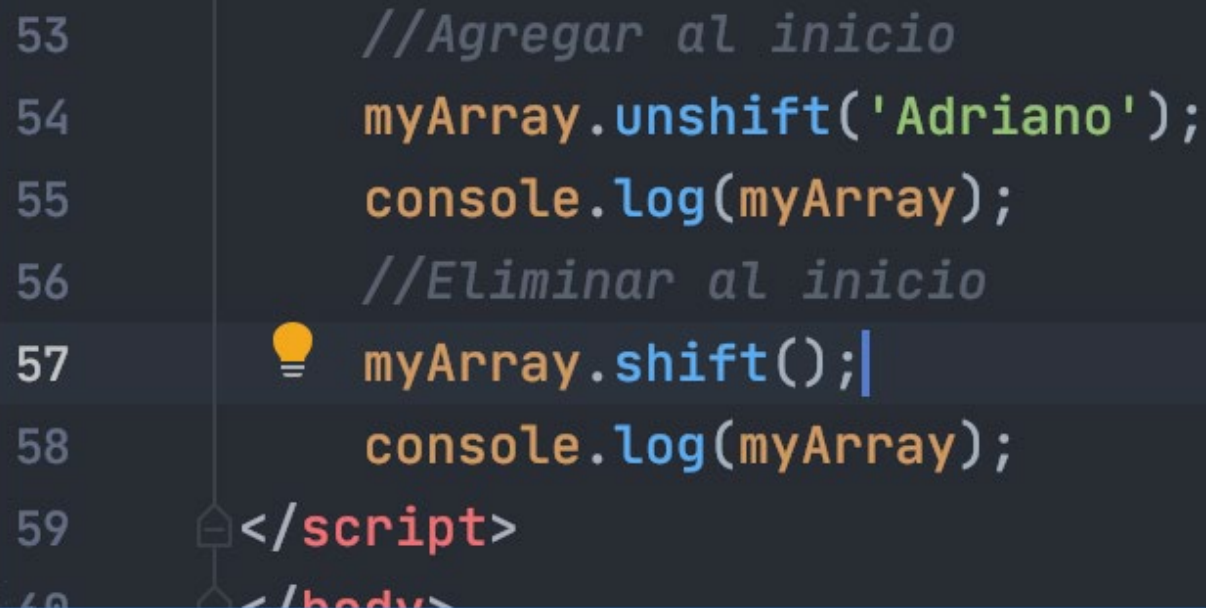
Antes que nada, para añadir o eliminar un elemento al final de una matriz podemos usar [push\(\)](#) y [pop\(\)](#) respectivamente




```
47 //Remover Item
48 console.log(myArray.pop());
49 let removedItem = myArray.pop();
50 console.log(myArray);
51 console.log(removedItem);
```

# Agregar o eliminar elementos de la matriz

[unshift\(\)](#) y [shift\(\)](#) funcionan exactamente igual de `push()` y `pop()`, respectivamente, excepto que funcionan al principio de la matriz, no al final.

A screenshot of a code editor window with a blue border. The editor has a dark background and shows JavaScript code. Line 53 has a comment `//Agregar al inicio`. Line 54 has `myArray.unshift('Adriano');`. Line 55 has `console.log(myArray);`. Line 56 has a comment `//Eliminar al inicio`. Line 57 has a lightbulb icon followed by `myArray.shift();`. Line 58 has `console.log(myArray);`. Line 59 has `</script>`. Line 60 has `</body>`.

```
53 //Agregar al inicio
54 myArray.unshift('Adriano');
55 console.log(myArray);
56 //Eliminar al inicio
57  myArray.shift();
58 console.log(myArray);
59 </script>
60 </body>
```

#MeGustaProgramar

# Gracias por su atención

Ing. Christian Arévalo Jesús

