Bucle For

for for for for for for for for for

for for for for for for for for ++

aprendé programando Virtual

SECUNDARIA del futuro

UTNFra

GyE Gestión y Evaluación Educativa

For

Nos permite crear un bucle, es decir repetir un código una y otra vez.
Se utiliza para realizar tareas que se repiten una y otra vez.

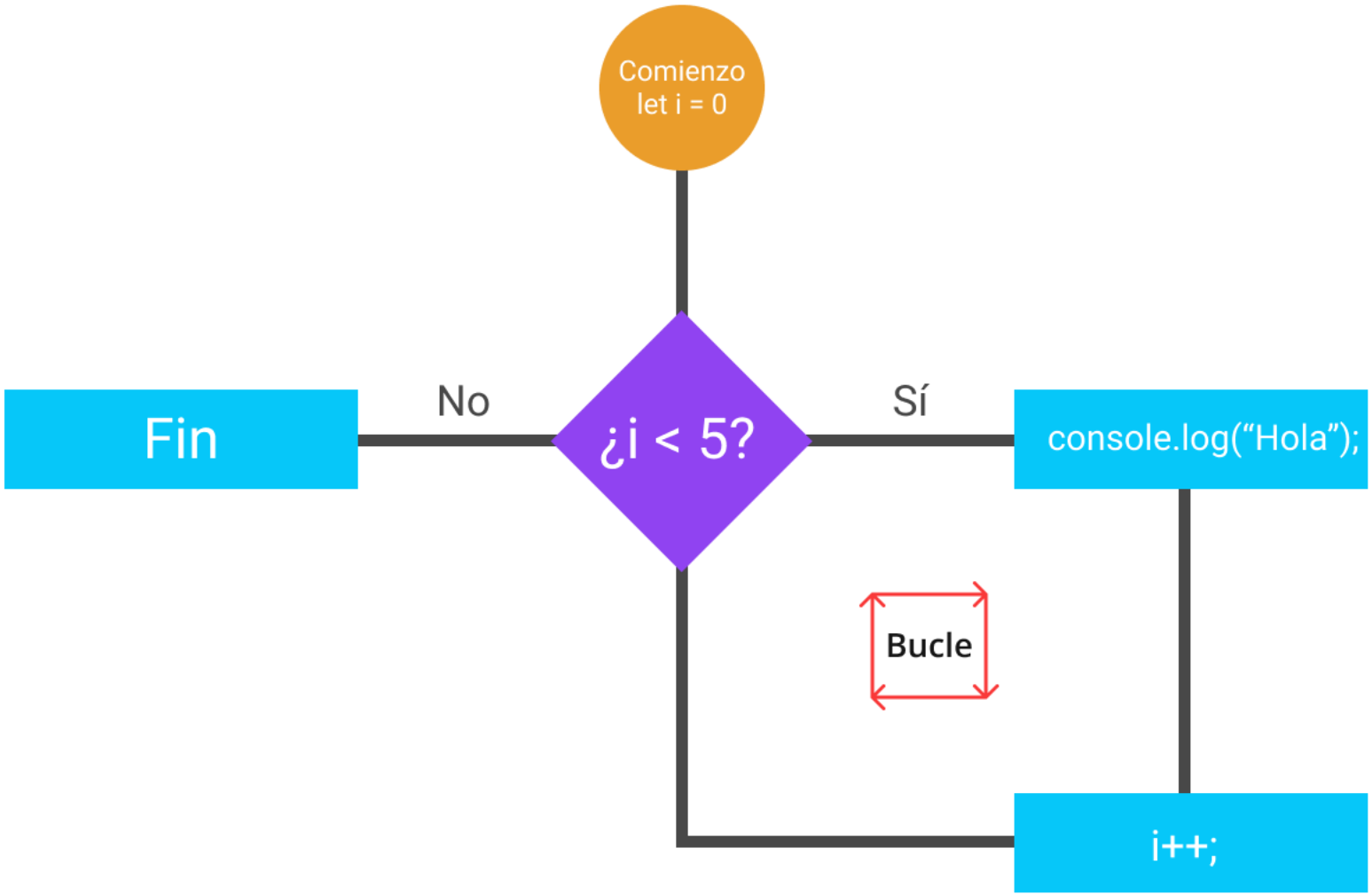
Utiliza la palabra reservada **for** y recibe 3 expresiones dentro de los paréntesis

for ([expresion-inicial]; [condicion]; [expresion-final]) { código a ejecutar }

expresión inicial: variable inicial
condición: condición de corte
expresión final: acción de incremento

For

Para entender el ciclo for, vamos a pensar primero en un diagrama:



For

Todas las expresiones juntas, escritas en código se ven de la siguiente manera:

```
for ( let i = 0; i < 5; i ++) {
    código a ejecutar
}
```

expresión inicial: definimos una variable i que comienza en 0

condición: definimos la condición de corte, en este caso se debe cumplir que i sea menor que 5

expresión final: luego de cada vuelta, se ejecuta esta parte del código, que en este caso sumamos 1 a la variable i

For y arrays

Además de utilizar el bucle for para repetir código, una de las funciones más importantes es la de poder recorrer arrays.

Tenemos el siguiente array:

```
let nombres = [“Juan”, “Mercedes”, “Sofía”, “Lucas”, “Luca”];
                0        1         2        3        4
```

y el siguiente bucle:

```
for ( let i = 0; i < 5; i ++) {
   código a ejecutar
}
```

¿Qué pasa si hacemos lo siguiente?

```
for ( let i = 0; i < 5; i ++) {
   console.log('Hola ' + nombres[i]);
}
```

Hola Juan
Hola Mercedes
Hola Sofía
Hola Lucas
Hola Luca

Veremos lo siguiente en la consola

For y arrays

Analicemos qué es lo que pasó

```
let nombres = [“Juan”, “Mercedes”, “Sofía”, “Lucas”, “Luca”];
                0         1          2        3        4
```

Viendo este for, sabemos que se va a ejecutar 5 veces.

```
for ( let i = 0; i < 5; i ++) {
   console.log(‘Hola ’ + nombres[i]);
}
```

1er Vuelta	2da Vuelta	3er Vuelta	4ta Vuelta	5ta Vuelta
i = 0	**i = 1**	**i = 2**	**i = 3**	**i = 4**
nombres[0] = “Juan”	***nombres[1] = “Mercedes”***	***nombres[2] = “Sofía”***	***nombres[3] = “Lucas”***	***nombres[4] = “Luca”***

La variable i se reemplaza por el valor que tiene en cada vuelta

For y arrays

Evaluando el array en [i], siendo **i** la variable que va cambiando, podemos recorrer un array por medio de un bucle for.

Ahora... ¿qué pasa si cambiamos la condición de corte, y en vez de i < 5, ponemos i < 6?

```
for ( let i = 0; i < 6; i ++) {
   console.log('Hola ' + nombres[i]);
}
```

1er Vuelta	2da Vuelta	3er Vuelta	4ta Vuelta	5ta Vuelta	6ta Vuelta
i = 0	**i = 1**	**i = 2**	**i = 3**	**i = 4**	**i = 5**
nombres[0] = "Juan"	***nombres[1] = "Mercedes"***	***nombres[2] = "Sofía"***	***nombres[3] = "Lucas"***	***nombres[4] = "Luca"***	***nombres[5] = undefined***

Como el array solo tenía 5 posiciones, si quiere buscar la 6ta, da como resultado "**undefined**" o que **no está definido**.

For y arrays

Para evitar este tipo de problemas, se usa la propiedad de los arrays .length que nos devuelve el número de elementos que tiene el array. De esta forma si el array cambia, el bucle queda actualizado siempre.

```
let nombres = [“Juan”, “Mercedes”, “Sofía”, “Lucas”, “Luca”];
                0        1          2        3        4
```

```
for ( let i = 0; i < nombres.length; i ++) {
   console.log(‘Hola ’ + nombres[i]);
}
```

El .length va a evitar que el bucle se exceda del número de elementos, cosa que puede pasar si lo escribimos a mano.

1er Vuelta	2da Vuelta	3er Vuelta	4ta Vuelta	5ta Vuelta
i = 0	**i = 1**	**i = 2**	**i = 3**	**i = 4**
nombres[0] = “Juan”	***nombres[1] = “Mercedes”***	***nombres[2] = “Sofía”***	***nombres[3] = “Lucas”***	***nombres[4] = “Luca”***