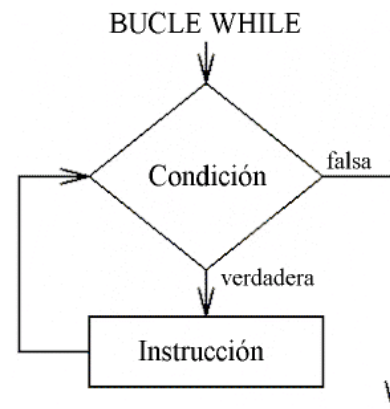
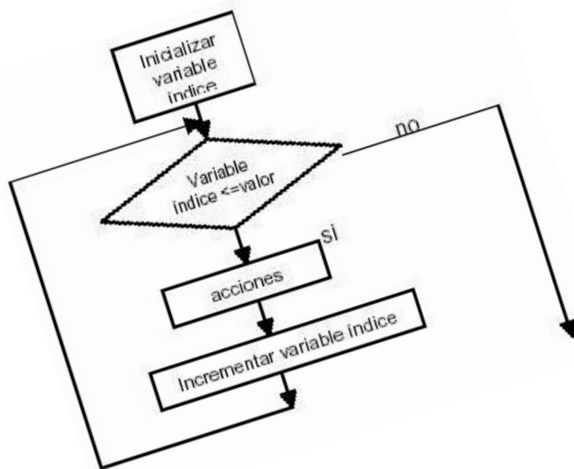


Bloque1: Programaci n b sica con JavaScript (JS)

Flujos/ Estructuras de control Repetitivas



sentencia while

```
while ( CONDICIÓN ) {
```


```
    _____  
    _____  
    _____  
    _____  
    _____  
}
```

ESTE CICLO ES MUY ÚTIL, CUANDO NO CONOCEMOS EXACTAMENTE LA CANTIDAD DE VECES QUE NECESITAMOS EJECUTAR EL CÓDIGO

CICLOS O LOOPS

Nos permiten ejecutar las mismas líneas de código una y otra vez de forma consecutiva

```
1  n = 0;  
2  x = 0;  
3  while (n < 3) {  
4      n++;  
5      x += n;  
6  }
```

- 
- Después del primer paso: `n = 1` y `x = 1`
 - Después del segundo paso: `n = 2` y `x = 3`
 - Después del tercer paso: `n = 3` y `x = 6`

sentencia do...while

Variante del **WHILE**, donde la condición está al final del bloque de código.

```
do  
    sentencia  
while (condicion);
```

```
let readlnSync = require('readline-sync');  
  
let salir = false;  
let respuesta;  
do {  
    respuesta = readlnSync.question('Deseas continuar (S/N)? ');  
    if (respuesta == 'N' || respuesta == 'n')  
        salir = true;  
} while (salir == false);
```

```
1 | do {  
2 |     i += 1;  
3 |     console.log(i);  
4 | } while (i < 5);
```

```
var num = 0;  
  
do{  
    num = num + 1;  
}while(num < 5);
```

Diferencia entre **while()** y **do..while()**



sentencia for

for

Es muy útil cuando sabemos la cantidad de veces que necesitamos ejecutar el código

for (let variable = 1; variable <= 10; variable ++) {

PUEDA SER CUALQUIER VALOR INICIAL

LA MISMA VARIABLE

SE EJECUTARÁ 10 VECES

Y EN CADA REPETICIÓN LA VARIABLE IRA INCREMENTANDO EN 1

}

CICLOS O LOOPS

Nos permiten ejecutar las mismas líneas de código una y otra vez de forma consecutiva

Ejemplo

```
1 var paso;  
2 for (paso = 0; paso < 5; paso++) {  
3   // Se ejecuta 5 veces, con valores desde paso desde 0 hasta 4.  
4   console.log('Dando un paso al Este');  
5 };
```

```
function howMany(selectObject) {  
  var numberSelected = 0;  
  for (var i = 0; i < selectObject.options.length; i++) {  
    if (selectObject.options[i].selected) {  
      numberSelected++;  
    }  
  }  
  return numberSelected;  
}
```

Ejemplo

Uso de sentencias BREAK y CONTINUE:

- **break:** Fuerza la salida dentro de un bucle (también en el switch)
- **continue:** Fuerza pasar a la siguiente iteración en un bucle.

```
var i = 0;
while (i < 6) {
    if (i == 3)
        break;
    i++;
}
```

Ejemplo

```
1  i = 0;
2  n = 0;
3  while (i < 5) {
4      i++;
5      if (i == 3) {
6          continue;
7      }
8      n += i;
9  }
```

Ejemplo

Uso de sentencias FOR..IN y FOR..OF:

for..in: itera una variable especificada sobre el nombre de las propiedades enumerables de un objeto.

```
for (variable en objeto) {  
    sentencias  
}
```

for..of: itera una variable especificada sobre los valores de las propiedades enumerables de un objeto.

```
for (variable de objeto) {  
    sentencia  
}
```

Llegado el momento,
las
usaremos
cuando
trabajemos
con objetos

```
1  let arr = [3, 5, 7];  
2  arr.foo = "hello";  
3  
4  for (let i in arr) {  
5      console.log(i); // logs "0", "1", "2", "foo"  
6  }  
7  
8  for (let i of arr) {  
9      console.log(i); // logs "3", "5", "7"  
10 }
```

Ejemplo

JS repetitivas1.js ●

```
1 // Programa que muestre del 1 al 10.  
2 // Realicemos tres versiones (con FOR, WHILE, DO..WHILE)
```

A practicar

```
4 let contador;  
5  
6 // Con for  
7 for (contador=1; contador<=10; contador++) {  
8     console.log('Valor de contador:' + contador);  
9 }  
10  
11 // con while  
12 contador = 1;  
13 while (contador<=10) {  
14     console.log('Valor de contador:' + contador);  
15     contador++; // igual que contador = contador + 1  
16 }  
17  
18 // con do..while  
19 contador = 1;  
20 do {  
21     console.log('Valor de contador:' + contador);  
22     contador++; // igual que contador = contador + 1  
23 } while (contador<=10);
```


Realicemos otros ejemplos...

JS `tabladel5.js` ✕

```
1 // Programa que realiza la tabla de multiplicar del 5.  
2 // 5 x 1 = 5  
3 // 5 x 2 = 10  
4 // ...  
5 // ...  
6 // 5 x 10 = 50
```

JS `tabladelX.js` ✕

```
1 // Programa que realiza la tabla de multiplicar de un número  
2 // suministrado en una variable.
```

JS `tabladelX.js` ●

```
1 // Programa que realiza la tabla de multiplicar de un número  
2 // suministrado en una variable. Ahora de forma decreciente (10 al 1)
```

Te atreves... ¿con for, while y do..while?

| 1x | 2x | 3x | 4x | 5x |
|-------------|-------------|-------------|-------------|---------------|
| 1 x 1 = 1 | 2 x 1 = 2 | 3 x 1 = 3 | 4 x 1 = 4 | 5 x 1 = 5 |
| 1 x 2 = 2 | 2 x 2 = 4 | 3 x 2 = 6 | 4 x 2 = 8 | 5 x 2 = 10 |
| 1 x 3 = 3 | 2 x 3 = 6 | 3 x 3 = 9 | 4 x 3 = 12 | 5 x 3 = 15 |
| 1 x 4 = 4 | 2 x 4 = 8 | 3 x 4 = 12 | 4 x 4 = 16 | 5 x 4 = 20 |
| 1 x 5 = 5 | 2 x 5 = 10 | 3 x 5 = 15 | 4 x 5 = 20 | 5 x 5 = 25 |
| 1 x 6 = 6 | 2 x 6 = 12 | 3 x 6 = 18 | 4 x 6 = 24 | 5 x 6 = 30 |
| 1 x 7 = 7 | 2 x 7 = 14 | 3 x 7 = 21 | 4 x 7 = 28 | 5 x 7 = 35 |
| 1 x 8 = 8 | 2 x 8 = 16 | 3 x 8 = 24 | 4 x 8 = 32 | 5 x 8 = 40 |
| 1 x 9 = 9 | 2 x 9 = 18 | 3 x 9 = 27 | 4 x 9 = 36 | 5 x 9 = 45 |
| 1 x 10 = 10 | 2 x 10 = 20 | 3 x 10 = 30 | 4 x 10 = 40 | 5 x 10 = 50 |
| 6x | 7x | 8x | 9x | 10x |
| 6 x 1 = 6 | 7 x 1 = 7 | 8 x 1 = 8 | 9 x 1 = 9 | 10 x 1 = 10 |
| 6 x 2 = 12 | 7 x 2 = 14 | 8 x 2 = 16 | 9 x 2 = 18 | 10 x 2 = 20 |
| 6 x 3 = 18 | 7 x 3 = 21 | 8 x 3 = 24 | 9 x 3 = 27 | 10 x 3 = 30 |
| 6 x 4 = 24 | 7 x 4 = 28 | 8 x 4 = 32 | 9 x 4 = 36 | 10 x 4 = 40 |
| 6 x 5 = 30 | 7 x 5 = 35 | 8 x 5 = 40 | 9 x 5 = 45 | 10 x 5 = 50 |
| 6 x 6 = 36 | 7 x 6 = 42 | 8 x 6 = 48 | 9 x 6 = 54 | 10 x 6 = 60 |
| 6 x 7 = 42 | 7 x 7 = 49 | 8 x 7 = 56 | 9 x 7 = 63 | 10 x 7 = 70 |
| 6 x 8 = 48 | 7 x 8 = 56 | 8 x 8 = 64 | 9 x 8 = 72 | 10 x 8 = 80 |
| 6 x 9 = 54 | 7 x 9 = 63 | 8 x 9 = 72 | 9 x 9 = 81 | 10 x 9 = 90 |
| 6 x 10 = 60 | 7 x 10 = 70 | 8 x 10 = 80 | 9 x 10 = 90 | 10 x 10 = 100 |

Recuerda:
***A PROGRAMAR
SE APRENDE...
PROGRAMANDO***



ACTIVIDAD PRÁCTICA 2 (AP2)

Título

Ejercicios básicos de programación (hasta estructuras de control repetitivas)

Objetivos

- Saber utilizar los tipos básicos del lenguaje JavaScript.
- Usar correctamente los operadores aritméticos, precedencias, operadores relacionales y lógicos.
- Saber utilizar las estructuras condicionales y repetitivas de un lenguaje de programación.

Temporalización

Previsto: Utilización de dos sesiones presenciales + trabajo en casa.