LENGUAJES DE SCRIPT DE CLIENTE IV

Ya hemos comentado que un bucle nos permite automatizar un conjunto de acciones que se deban realizar de forma repetitiva con el fin de ahorrar código. Ya conocemos el bucle for, que se ejecutará un número determinado de veces, condicionado por los valores que puede tomar su índice.

Hay otros tipos de bucles, que no se sabe cuántas veces se pueden ejecutar. Son los siguientes:

Bucle While.

Un bucle **while** es una sentencia de control de flujo que permite que el código se ejecute repetidamente en función de una condición dada. Mientras la condición sea cierta, el cuerpo del bucle seguirá ejecutándose. En cada paso de ejecución, se vuelve a comprobar la condición. Por tanto, la condición ha de ser susceptible de ser modificada dentro del cuerpo del bucle ya que si no, será un bucle infinito.

Sintaxis:

```
while (condición)
{
  Instrucciones que se ejecutan dentro del bucle
}
```

- El **while** comienza con la verificación de la condición. Si se evalúa como verdadero, las instrucciones del cuerpo del bucle se ejecutan; de lo contrario, el cuerpo del bucle nunca se ejecutaría. Por esta razón, también se llama **bucle de control de entrada**.
- Una vez que la condición se evalúa como verdadera, se ejecutan las instrucciones en el cuerpo del bucle.
- Normalmente, las declaraciones contienen un valor de actualización para la variable que se procesa para la siguiente iteración.
- Cuando la condición se vuelve falsa, el bucle finaliza y la ejecución del programa continúa por la siguiente instrucción debajo del bucle.

Ejemplo:

```
int x = 1;
// Salir cuando x llega a ser mayor que 4
while (x <= 5)
{
    document.write("Valor de x: " + x);
    //incrementa el valor de x para la siguiente iteración
    x=x+1;
}
```

La salida que genera la ejecución de este bucle sería:

Valor de x: 1
Valor de x: 2
Valor de x: 3
Valor de x: 4
Valor de x: 5

Bucle Do .. While.

El bucle **do.. while** es similar al **while** con la única diferencia de que comprueba la condición después de ejecutar las instrucciones, y por lo tanto, sabemos que como mínimo, se va a ejecutar una vez el cuerpo del bucle.

Sintaxis:

```
do {
  Instrucciones que se ejecutan dentro del bucle
} while (condición);
```

- El bucle **do while** comienza con la ejecución de las instrucciones del cuerpo del bucle. No hay verificación de ninguna condición la primera vez.

- Después de la ejecución del cuerpo y la actualización del valor de la variable que interviene en la condicion, la condición se verifica para el valor verdadero o falso. Si se evalúa como verdadero, comienza la siguiente iteración del ciclo.

- Cuando la condición se vuelve falsa, el ciclo finaliza y el bucle termina.

- Es importante tener en cuenta que el **bucle do-while ejecutará sus declaraciones al menos una vez antes de que se verifique cualquier condición.**

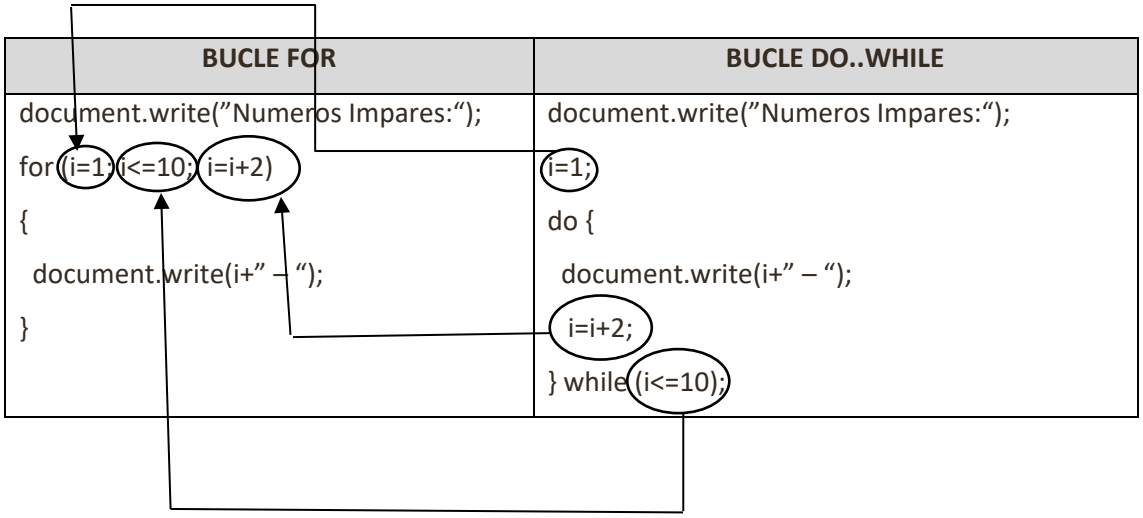
Vamos a poner un ejemplo explicado:

Programa que imprime los números pares del 1 al 10

```
var i;
//Inicializamos la variable que controla el bucle
 i=1;
 do {
  if (i%2==0)
    document.write (i+" - ");
    //incrementamos en uno el valor de i. Si no modificásemos este valor dentro del cuerpo
    //del bucle, éste sería infinito, pues siempre valdría lo mismo y la condición nunca cambiaría
  i=i+1;
 } while (i<=100);   //Mientras i sea menor o igual a 100, el bucle ha de seguir ejecutándose
```

Por tanto, vemos que un bucle FOR puede ser emulado por un DO .. WHILE.

Veamos cómo con un ejemplo (fijaros que los datos que se utilizan son los mismos, solo que dispuestos de otra manera).



BUCLE FOR	BUCLE DO..WHILE
document.write("Numeros Impares:");	document.write("Numeros Impares:");
for(i=1;i<=10; i=i+2)	i=1;
{	do {
document.write(i+" – ");	document.write(i+" – ");
}	i=i+2;
	} while(i<=10);

Ejercicios.

1. Programa que pida un numero por teclado y dibuje una fila centrada de tantos asteriscos como indique el número. Si el número es par los pintará en azul si es impar, en rojo. Realizarlo con bucle WHILE.
2. Pedir un numero por teclado y que muestre su tabla de multiplicar (del 1 al 10), centrada y como se muestra en el grafico. Realizarlo con DO..WHILE.

Tabla de multiplicar del 8				
8	x	1	=	8
8	X	2	=	16
8	X	3	=	24
.....	....	....	....	.....
8	x	10	=	80