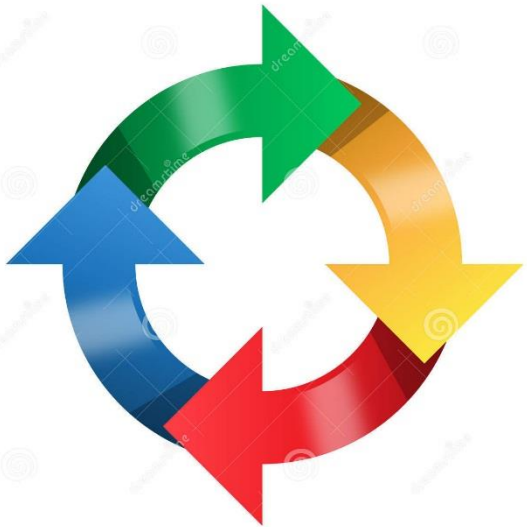


SENTENCIAS DE REPETICIÓN EN C#

CICLOS



```
do {  
    cerveza++;  
} while (sobrio) ;
```

Los ciclos repetitivos también llamados lazos o bucles permiten repetir una operación o secuencia de operaciones en función de ciertas condiciones. Es un segmento de un algoritmo o programa cuyas instrucciones se repiten un número determinado de veces mientras se cumpla una determinada condición.

Dentro de los ciclos se utilizan contadores y acumuladores, que regulan que el ciclo llegue a su fin.

CICLOS

- Las estructuras de control iterativas se clasifican en tres tipos:
- **Para: (for)** se utiliza cuando se puede determinar el número de veces que hay que ejecutar las instrucciones.
- **Mientras: (While)** - el ciclo mientras es la estructura básica que permite repetir varias veces una secuencia de operaciones, mientras se cumpla una determina condición.
- **Hacer - Mientras: (Do-While)** - las operaciones de lazo DO son ejecutadas mientras la condición sea cierta.

Contador

- Es un tipo de variable que incrementa o decrementa su valor en un valor constante:

```
int x=1;
```

- `X++;`
- `X--;`
- `X+=2;`

Cuando decimos incrementa estamos sumando.

Ejemplo: `Veces = Veces + 1`

Como se puede observar a la variable `veces` se le está incrementando un valor constante (1); es decir a su contenido le sumas el valor y se vuelve a guardar en la misma variable.

Acumulador

- Es una variable que incrementa o decrementa su contenido en cantidades variables.

Ejemplo

Nomina = Nomina + sueldo



CICLO FOR

CICLO FOR

- El ciclo for es un ciclo muy flexible y a la vez muy potente ya que tiene varias formas interesantes de implementarlo.

En C SHARP este ciclo es uno de los mas usados para repetir una secuencia de instrucciones sobre todo cuando **se conoce la cantidad exacta de veces** que se quiere que se ejecute una instrucción simple o compuesta.

```
for (<inicialización>; <condición>; <incremento>)  
{  
    //código a ejecutar  
}
```

- En su forma simple la inicialización es una instrucción de asignación que carga una variable de control de ciclo con un valor inicial.
- La condición es una expresión relacional que evalúa la variable de control de ciclo contra un valor final o de parada que determina cuando debe acabar el ciclo.
- El incremento define la manera en que la variable de control de ciclo debe cambiar cada vez que el computador repite un ciclo.
- Se deben separar esos 3 argumentos con punto y coma (;)

Estructura
Iterativa

Inicialización

Expresión Booleana

Actualización

```
for (int i=1; i<=12; i++) {  
    suma= num + i;  
}
```

Instrucciones que se
repetirán

CICLO FOR

```
for (int i=0; i < 100; i++)  
{  
    //Código a ejecutar  
}
```

Es importante comentar que esos tres "parámetros" que se le pasan a la sentencia for se pueden omitir, pero los punto-y-coma entre ellos sí se han de dejar. También cabe destacar que si no se pone la condición de ejecución del ciclo, este se ejecutará indefinidamente hasta que se ejecute una sentencia break o se termine la función o el programa mediante un return.

EJEMPLOS

```
for(int i=0 ; i<=10; i++)  
{  
    MessageBox.Show("Numero = "+i);  
}
```

Sentencias (4)

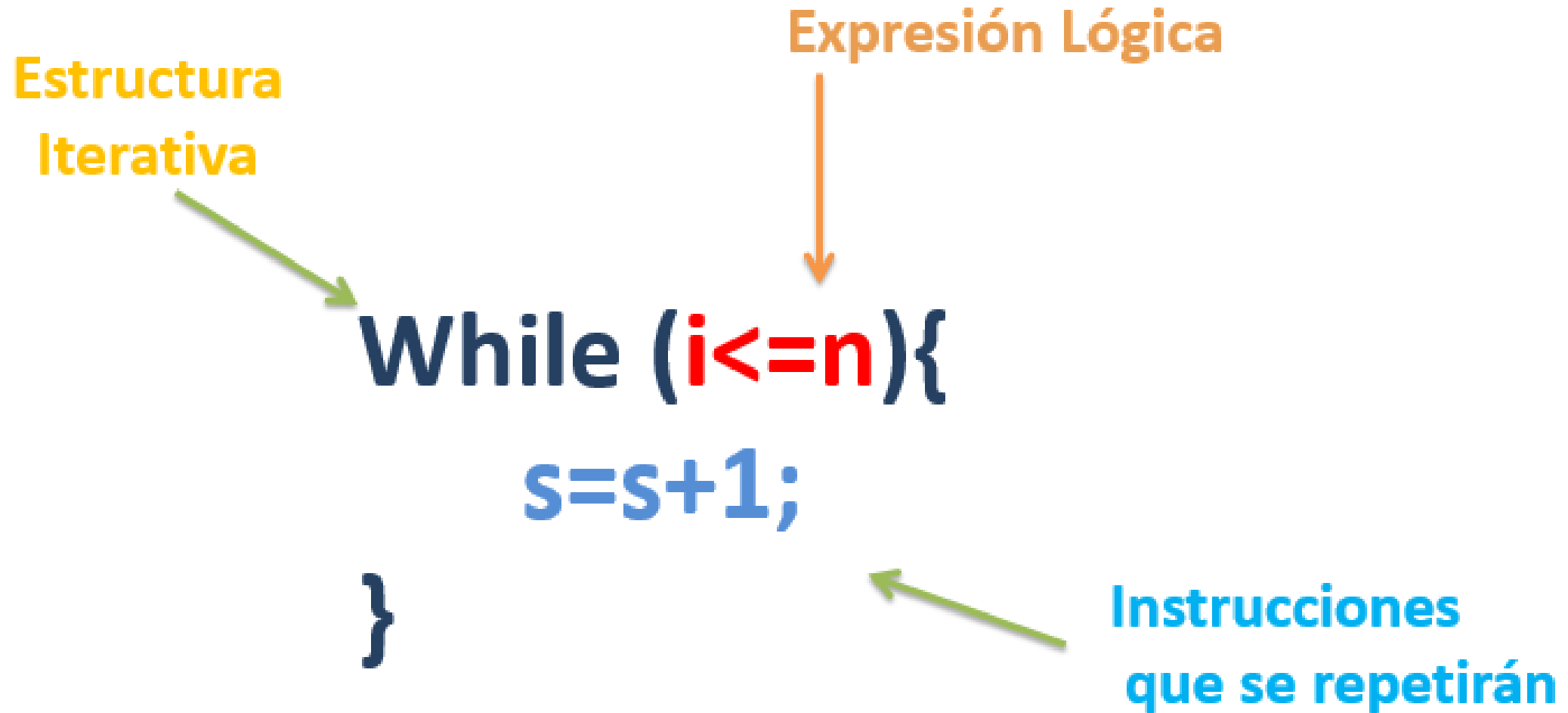
⇒ Instrucción WHILE:

Sirve repetir una serie de instrucciones mientras se cumpla una condición.

```
while (<condición>)  
{  
    <instrucciones>  
}
```

Estructura
Iterativa

Expresión Lógica



```
While (i<=n){  
    s=s+1;  
}
```

The diagram illustrates the components of a while loop. The text 'While (i<=n){' is in dark blue, 's=s+1;' is in light blue, and the closing brace '}' is in dark blue. An orange arrow points from 'Expresión Lógica' to the condition '(i<=n)'. A green arrow points from 'Estructura Iterativa' to the 'While' keyword. Another green arrow points from 'Instrucciones que se repetirán' to the body of the loop 's=s+1;'.

Instrucciones
que se repetirán

Estructura
Iterativa

do{

s=s+1;

}

While (nota<=0 || nota>=20)

Instrucciones
que se repetirán

Expresión Lógica

Sentencias (5)

⇒ Instrucción DO WHILE:

Sirve repetir una serie de instrucciones mientras se cumpla una condición y al menos se ejecutan 1 vez.

```
do
{
    <instrucciones>
}
while (<condición>)
```