CICLOS 1 JAVA

¿Qué es un ciclo (o bucle)?

Imagina que necesitas enviarle el mismo mensaje a 100 personas. ¿Escribirías el mensaje 100 veces? ¡Por supuesto que no! Usarías "copiar y pegar". En programación, un **ciclo** (o bucle) es exactamente eso: una forma de **repetir una o más instrucciones** un número determinado de veces o hasta que se cumpla una condición específica. Es una de las herramientas más poderosas y fundamentales en la programación.

¿Para qué se usa un ciclo?

Los ciclos son esenciales para automatizar tareas repetitivas. Sin ellos, el código sería increíblemente largo y difícil de mantener. Se usan para:

- **Recorrer colecciones de datos**: Como listas (Arrays), donde necesitas realizar la misma operación en cada elemento.
- **Realizar operaciones un número fijo de veces**: Por ejemplo, "imprimir los números del 1 al 10".
- **Leer datos hasta que se cumpla una condición**: Como leer la entrada de un usuario hasta que escriba "salir".
- **Crear simulaciones o animaciones**: Donde cada "paso" de la simulación es una repetición del ciclo.

El Ciclo for

El ciclo for es ideal cuando **sabes de antemano cuántas veces** quieres que se repita una acción. Su estructura es muy clara y se divide en tres partes clave, separadas por punto y coma ;.

Sintaxis básica:

Java

```
for (inicialización; condición; actualización) {
    // Código que se repetirá
}
```

- **Inicialización**: Se ejecuta **una sola vez** al principio. Aquí es donde declaras e inicializas una variable de control (generalmente llamada i por "índice"). int i = 0;

- **Condición**: Antes de cada repetición (incluida la primera), se evalúa esta condición. Si es **verdadera (true)**, el código dentro del ciclo se ejecuta. Si es **falsa (false)**, el ciclo termina. i < 10;
- **Actualización**: Se ejecuta **al final de cada repetición**. Generalmente se usa para incrementar o decrementar la variable de control. i++ (incrementa i en 1).

Ejemplos de Uso del Ciclo for

Ejemplo Básico: Contar del 1 al 10

Este es el "Hola, Mundo" de los ciclos. Simplemente imprimimos los números del 1 al 10.

Java

```java
// Objetivo: Imprimir números del 1 al 10.
System.out.println("Iniciando conteo:");

for (int i = 1; i <= 10; i++) {
    System.out.println("Número: " + i);
}

System.out.println("¡Conteo finalizado!");
```

Explicación:

1. **int i = 1**: Empezamos nuestra variable i en 1.
2. **i <= 10**: El ciclo continuará mientras i sea menor o igual a 10.
3. **i++**: Después de cada impresión, i aumenta su valor en 1.

Ejemplo Intermedio: Sumar los primeros 50 números pares

Aquí, usaremos el ciclo para realizar un cálculo acumulativo.

Java

```java
// Objetivo: Calcular la suma de los primeros 50 números pares (2,
4, 6...).
int sumaTotal = 0;

for (int i = 2; i <= 100; i += 2) {
    sumaTotal = sumaTotal + i; // Acumula el valor de i en
```

```
sumaTotal
}

System.out.println("La suma de los primeros 50 números pares es: "
+ sumaTotal);
```

Explicación:

1. **int i = 2**: Empezamos en 2, el primer número par.
2. **i <= 100**: El número par 50º es 100, así que esa es nuestra condición de parada.
3. **i += 2**: En lugar de i++, incrementamos de 2 en 2 para saltar solo a los números pares.

Ejemplo Avanzado: Generar la serie de Fibonacci

La serie de Fibonacci es una secuencia donde cada número es la suma de los dos anteriores (0, 1, 1, 2, 3, 5, 8...). Usaremos un ciclo for para generar los primeros 15 términos.

Java

```
// Objetivo: Generar los primeros 15 términos de la serie de
Fibonacci.
int n = 15; // Cantidad de términos a generar
int a = 0, b = 1;

System.out.print("Serie de Fibonacci (" + n + " términos): ");
System.out.print(a + " " + b + " ");

for (int i = 2; i < n; i++) {
    int siguiente = a + b;
    System.out.print(siguiente + " ");
    a = b;
    b = siguiente;
}

System.out.println(); // Salto de línea al final
```

Explicación:

1. **int i = 2**: Empezamos en 2 porque ya hemos impreso los dos primeros términos (0 y 1).

2. **i < n**: El ciclo se repite hasta que hayamos generado n términos en total.
3. Dentro del ciclo, calculamos el **siguiente** término, lo imprimimos y luego **actualizamos** las variables a y b para prepararlas para la siguiente repetición.

Taller Final: Domina el Ciclo for

Dificultad Básica

Contador descendente: Escribe un programa que imprima los números del 10 al 1 en orden descendente.

Tabla de multiplicar: Pide al usuario un número e imprime su tabla de multiplicar del 1 al 10. (Ej: Si ingresa 7, imprime "7 x 1 = 7", "7 x 2 = 14", ...).

Números impares: Imprime todos los números impares entre 1 y 50.

Dificultad Media

Suma de un rango: Pide al usuario dos números (un inicio y un fin) y calcula la suma de todos los números en ese rango (incluyendo el inicio y el fin).

Calculadora de factorial: Pide un número al usuario y calcula su factorial (n!). El factorial de un número n es la multiplicación de todos los números enteros positivos desde 1 hasta n. (Ej: 5!=5×4×3×2×1=120).

Potencia sin Math.pow: Pide al usuario una base y un exponente, y calcula la potencia. (Ej: base=2, exponente=5 -> resultado=32). No puedes usar la función Math.pow().

Dificultad Avanzada

Detector de número primo: Pide al usuario un número entero positivo y determina si es un número primo. Un número primo es aquel que solo es divisible por 1 y por sí mismo. *Pista: Usa un ciclo para verificar si tiene algún divisor entre 2 y el número - 1*.

Dibujar una línea: Pide al usuario un número y dibuja una línea horizontal de asteriscos * de esa longitud. (Ej: si ingresa 8, imprime ********).

Calculadora de Máximo Común Divisor (MCD): Pide al usuario dos números y encuentra su Máximo Común Divisor. El MCD es el mayor número que divide a ambos sin dejar resto. *Pista: puedes usar un ciclo que vaya desde 1 hasta el menor de los dos números*.