

A blurred background image of a computer screen displaying code. The code is in a dark theme with a blue and orange color scheme. The visible code includes comments and assignments for a 'mirror_mod' object, such as 'Get mirror object to mirror_mod.mirror_object', 'operation == "MIRROR_X":', and 'mirror_mod.use_x = True'.

```
mirror_mod = modifier_ob.  
Get mirror object to mirror_mod.mirror_object  
mirror_mod.mirror_object  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False
```

Módulo 7

Ciclos

Ciclos

```
1 public class Loop5{
2     public static void main(String[] args){
3         System.out.println(1);
4         System.out.println(2);
5         System.out.println(3);
6         System.out.println(4);
7         System.out.println(5);
8     }
9 }
```

Diseña un programa que imprima los números del 1 al 5.

Diseña un programa que imprima los números del 1 al 10.

```
1 public class Loop10{
2     public static void main(String[] args){
3         System.out.println(1);
4         System.out.println(2);
5         System.out.println(3);
6         System.out.println(4);
7         System.out.println(5);
8         System.out.println(6);
9         System.out.println(7);
10        System.out.println(8);
11        System.out.println(9);
12        System.out.println(10);
13    }
14 }
```

```

1 public class Loop5{
2     public static void main(String[] args){
3         System.out.println(1);
4         System.out.println(2);
5         System.out.println(3);
6         System.out.println(4);
7         System.out.println(5);
8     }
9 }

```

```

1 public class Loop10{
2     public static void main(String[] args){
3         System.out.println(1);
4         System.out.println(2);
5         System.out.println(3);
6         System.out.println(4);
7         System.out.println(5);
8         System.out.println(6);
9         System.out.println(7);
1        System.out.println(8);
10       System.out.println(9);
11       System.out.println(10);
12    }
13 }

```

Diseña un programa que imprima los números del 1-100

```

public class Loop5{
    public static void main(String[] args){
        System.out.println(1);
        System.out.println(2);
        System.out.println(3);
        System.out.println(4);
        System.out.println(5);
        System.out.println(6);
        System.out.println(7);
        System.out.println(8);
        System.out.println(9);
    }
}

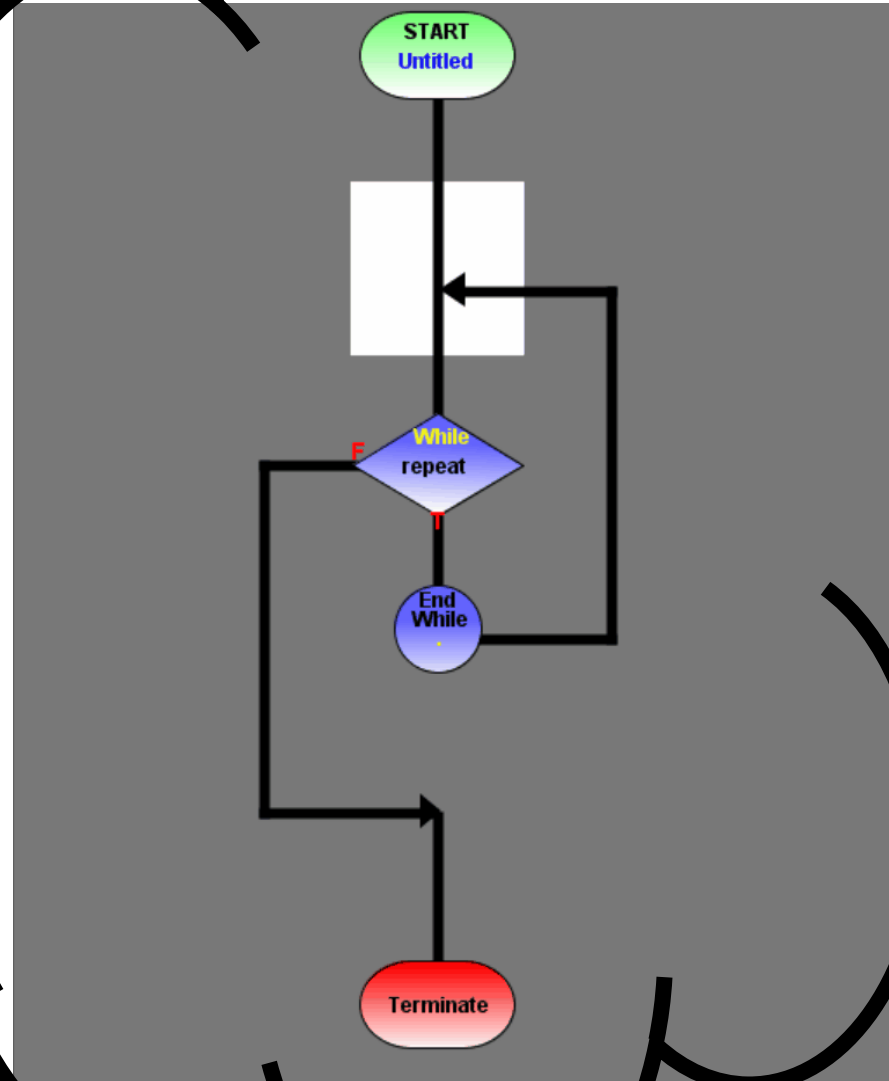
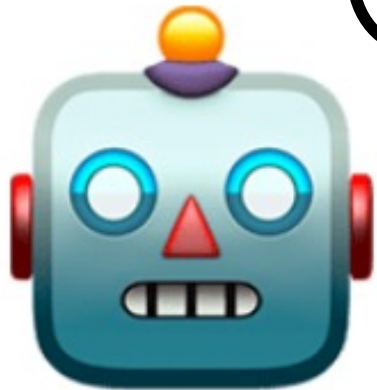
```

Diseña un programa que imprima los números del 1-100



```
public class Loop100{  
    public static void main(String[] args){  
        System.out.println(1);  
        System.out.println(2);  
        System.out.println(3);  
        System.out.println(4);  
        System.out.println(5);  
        System.out.println(6);  
        System.out.println(7);  
        System.out.println(8);  
        System.out.println(9);  
        System.out.println(10);  
        System.out.println(11);  
        System.out.println(12);  
        System.out.println(13);  
        System.out.println(14);  
        System.out.println(15);  
        System.out.println(16);  
        System.out.println(17);  
        System.out.println(18);  
        System.out.println(19);  
        System.out.println(20);  
        System.out.println(21);  
        System.out.println(22);  
        System.out.println(23);  
    }  
}
```

**¡Las computadoras
son muy buenas para
repetir instrucciones
al pie de la letra!**



Ciclos

Un ciclo es un bloque de instrucciones que van a repetirse, mientras se cumpla una condición.

1. **Cuerpo del ciclo:** ¿Qué quiero repetir?
2. **Variables de Control:** ¿Qué variable va a controlar la repetición?
3. **Condición de Salida:** ¿Cuántas veces los queremos repetir?

Ciclos while

Sintaxis

```
01 while(boolean_condition){  
02     //  
03     //code block to be executed  
04     //  
05 }
```

Ejemplos de condiciones:

```
01 i < 10  
02 keepGoing == true  
03 residuo != 0
```

Ejemplo: Identifica cada componente del ciclo:

- Control variable
- Body
- Exit condition

```
01 public class SampleLoop {
02     public static void main(String[] args) {
03         int bits = 1;
04         int maxValues = 1;
05         while (bits <= 8) {
06             maxValues *= 2;
07             System.out.println(bits + " bit(s) -> " + maxValues + " v
08             bits++;
09         }
10     }
11 }
```

```
01 public class SampleLoop {  
02     public static void main(String[] args) {  
03         int bits = 1;  
04         int maxValues = 1;  
05         while (bits <= 8) {  
06             maxValues *= 2;  
07             System.out.println(bits + " bit(s) -> " + maxValues + " v  
08             bits++;  
09         }  
10     }  
11 }
```

El ciclo completo abarca desde la línea 5 a 9.

```
01 public class SampleLoop {  
02     public static void main(String[] args) {  
03         int bits = 1;  
04         int maxValues = 1;  
05         while (bits <= 8) {  
06             maxValues *= 2;  
07             System.out.println(bits + " bit(s) -> " + maxValues + " v  
08             bits++;  
09         }  
10     }  
11 }
```

Variable de control: bits.

```
01 public class SampleLoop {  
02     public static void main(String[] args) {  
03         int bits = 1;  
04         int maxValues = 1;  
05         while (bits <= 8) {  
06             maxValues *= 2;  
07             System.out.println(bits + " bit(s) -> " + maxValues + " v  
08             bits++;  
09         }  
10     }  
11 }
```

Condición: repetir mientras bits sea menor o igual a 8.

```
01 public class SampleLoop {  
02     public static void main(String[] args) {  
03         int bits = 1;  
04         int maxValues = 1;  
05         while (bits <= 8) {  
06             maxValues *= 2;  
07             System.out.println(bits + " bit(s) -> " + maxValues + " v  
08             bits++;  
09         }  
10     }  
11 }
```

Cuerpo del ciclo.

```
01 public class SampleLoop {  
02     public static void main(String[] args) {  
03         int bits = 1;  
04         int maxValues = 1;  
05         while (bits <= 8) {  
06             maxValues *= 2;  
07             System.out.println(bits + " bit(s) -> " + maxValues + " v  
08             bits++;  
09         }  
10     }  
11 }
```

Condición de salida: bits mayor que 8

Prueba de Escritorio

initial values -->

bits	maxValues	out	bits <= 8
1	1		TRUE
2	2	1 bit(s) -> 2 values	TRUE
3	4	2 bit(s) -> 4 values	TRUE
4	8	3 bit(s) -> 8 values	TRUE
5	16	4 bit(s) -> 16 values	TRUE
6	32	5 bit(s) -> 32 values	TRUE
7	64	6 bit(s) -> 64 values	TRUE
8	128	7 bit(s) -> 128 values	TRUE
9	256	8 bit(s) -> 256 values	TRUE

<-- Enters while loop

<-- Exit condition is met

Ciclos do-while

Do-While

El do-while es tipo de ciclo, muy similar al while, con la excepción de que el procesamiento se va a ejecutar **por lo menos una vez**.

La sintaxis es la siguiente:

```
01  do {  
02      //  
03      //code block to be executed  
04      //  
05  } while (boolean_condition);
```

```
01    boolean invalidInput = true;
02    int i;
03    Scanner keyboard = new Scanner(System.in);
04    do {
05        System.out.print("Type a number between 1 and 10: ");
06        i = keyboard.nextInt();
07        if (i >= 1 && i <= 10){
08            invalidInput = false;
09        } else {
10            System.out.println("Error. Try again!");
11        }
12    } while(invalidInput == true);
13
14    System.out.println("Success: " + i + " is a valid number");
15
16    keyboard.close();
```

```
01     boolean invalidInput = true;
02     int i;
03     Scanner keyboard = new Scanner(System.in);
04     do {
05         System.out.print("Type a number between 1 and 10: ");
06         i = keyboard.nextInt();
07         if (i >= 1 && i <= 10){
08             invalidInput = false;
09         } else {
10             System.out.println("Error. Try again!");
11         }
12     } while(invalidInput == true);
13
14     System.out.println("Success: " + i + " is a valid number");
15
16     keyboard.close();
```

El ciclo abarca de la línea 4 a la 12.

```
01     boolean invalidInput = true;
02     int i;
03     Scanner keyboard = new Scanner(System.in);
04     do {
05         System.out.print("Type a number between 1 and 10: ");
06         i = keyboard.nextInt();
07         if (i >= 1 && i <= 10){
08             invalidInput = false;
09         } else {
10             System.out.println("Error. Try again!");
11         }
12     } while(invalidInput == true);
13
14     System.out.println("Success: " + i + " is a valid number");
15
16     keyboard.close();
```

El cuerpo del ciclo de la 5-11. Estas instrucciones se repetirán.

```
01     boolean invalidInput = true;
02     int i;
03     Scanner keyboard = new Scanner(System.in);
04     do {
05         System.out.print("Type a number between 1 and 10: ");
06         i = keyboard.nextInt();
07         if (i >= 1 && i <= 10){
08             invalidInput = false;
09         } else {
10             System.out.println("Error. Try again!");
11         }
12     } while(invalidInput == true);
13
14     System.out.println("Success: " + i + " is a valid number");
15
16     keyboard.close();
```

Condicion: Repetir mientras invalidInput almacene un valor de "true".

```
01     boolean invalidInput = true;
02     int i;
03     Scanner keyboard = new Scanner(System.in);
04     do {
05         System.out.print("Type a number between 1 and 10: ");
06         i = keyboard.nextInt();
07         if (i >= 1 && i <= 10){
08             invalidInput = false;
09         } else {
10             System.out.println("Error. Try again!");
11         }
12     } while(invalidInput == true);
13
14     System.out.println("Success: " + i + " is a valid number");
15
16     keyboard.close();
```

Condición de salida: Detener la repetición cuando invalidInput almacene el valor booleano "false".


```
Type a number between 1 and 10: 20
Error. Try again!
Type a number between 1 and 10: -15
Error. Try again!
Type a number between 1 and 10: 33
Error. Try again!
Type a number between 1 and 10: 5
Success: 5 is a valid number
```

Ciclos for

For

Los ciclos for generalmente son utilizados para repetir una serie de instrucciones una cantidad de veces fija.

La sintaxis es la siguiente:

```
01  for(initialization; boolean_condition; update){  
02      //  
03      // code block to be executed  
04      //  
05  }
```

```
01     for (int i = 9; i >= 0; i = i - 2) {  
02         System.out.println(i);  
03     }
```

```
01      for (int i = 9; i >= 0; i = i - 2) {  
02          System.out.println(i);  
03      }
```

Ciclo completo

```
01      for (int i = 9; i >= 0; i = i - 2) {  
02          System.out.println(i);  
03      }
```

`int i = 9` <-- Inicialización. Se declara una variable que sólo existe dentro del ciclo.

```
01         for (int i = 9; i >= 0; i = i - 2) {  
02             System.out.println(i);  
03         }
```

$i \geq 0$ <-- Condición. Mientras esta condición se cumpla, el ciclo for continuará ejecutándose.

```
01         for (int i = 9; i >= 0; i = i - 2) {  
02             System.out.println(i);  
03         }
```

$i = i - 2$ <-- Al final de cada ciclo, esta operación se ejecutará, reduciendo el valor de "i" en "2".

Ejemplo



Parrot Salute

En el zoológico de Monterrey hay un cotorro muy educado que saluda a cada grupo de personas que pasan. Lee del teclado el tamaño de un grupo, e imprime un saludo por cada persona.

Toma en cuenta lo siguiente:

- Los grupos no pueden ser de menos de 0 personas.
- Los grupos no pueden ser de más de 10 personas.

```
01     Scanner keyboard = new Scanner(System.in);
02
03     System.out.print("How big is your group?: ");
04     int groupSize = keyboard.nextInt();
05
06     while (groupSize <= 0 || groupSize >= 11){
07         System.out.print("Hmm that doesn't seem right. How big is your group: ");
08         groupSize = keyboard.nextInt();
09     }
10
11     String salute = "";
12     for(int i = 0; i < groupSize; i++){
13         salute = salute + "Hi";
14         if (i < (groupSize-1)){
15             salute = salute + "-";
16         }
17     }
18
19     System.out.println(salute);
```

Problemas Comunes

Errores de Lógica

Estos errores se originan por defectos inyectados por el programador. El programa no funciona de la manera esperada en alguno o todos los casos.

En los ciclos, hay errores como:

1. Off by one
2. Infinite loops
3. Wrong boolean expressions

Estos errores son más difíciles de encontrar, pues requiere que entendamos la intención, el código y el programa que estamos evaluando.

```
01 // off by one... prints only "hol"
02 String s1 = "hola";
03 int len = s1.length() - 1;
04 for (int i = 0; i < len; i++) {
05     System.out.print(s1.charAt(i));
06 }
```

```
01 //i is never updated, so this loops infinitely
02 while(i < 10){
03     System.out.println(i);
04 }
05
06 // wrong condition i<0, causing an infinite loop
07 for (int i = 10; i>0; i++){
08     System.out.println(i);
09 }
```