Módulo 7

Ciclos

Ciclos

```java
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);
    }
}
```

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

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

```java
public class Loop10{
    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);
    }
}
```

```java
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);
    }
}
```

```java
public class Loop10{
    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);
    }
}
```

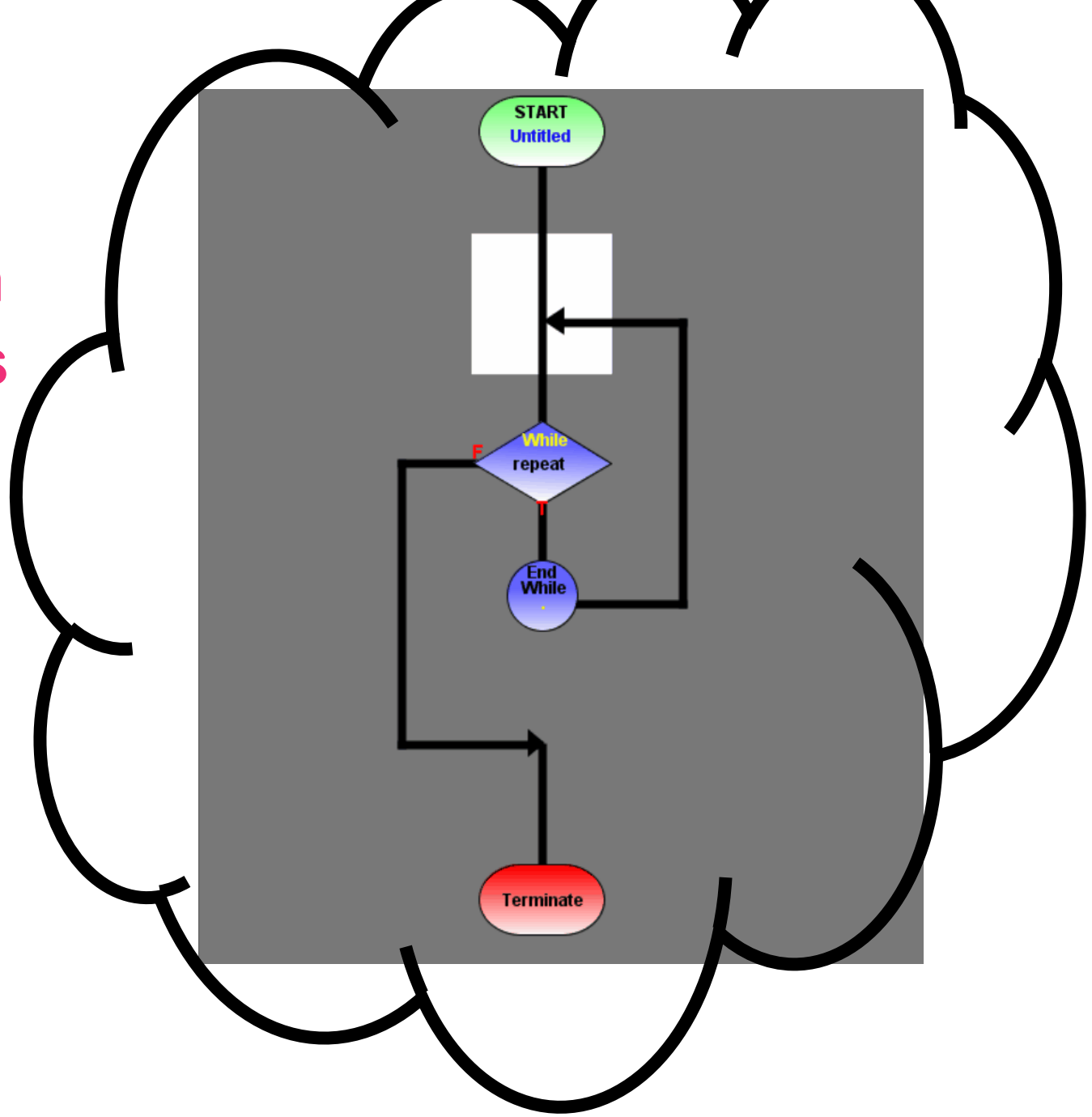
Diseña un programa que imprima los números del 1-100

```java
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

```java
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

```java
public class SampleLoop {
    public static void main(String[] args) {
        int bits = 1;
        int maxValues = 1;
        while (bits <= 8) {
            maxValues *= 2;
            System.out.println(bits + " bit(s) -> " + maxValues + " v
            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  }
```

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.

```java
public class SampleLoop {
    public static void main(String[] args) {
        int bits = 1;
        int maxValues = 1;
        while (bits <= 8) {
            maxValues *= 2;
            System.out.println(bits + " bit(s) -> " + maxValues + " v
            bits++;
        }
    }
}
```

Condición de salida: bits mayor que 8

Prueba de Escritorio

	bits	maxValues	out	bits <= 8	
initial values -->	1	1		TRUE	<-- Enters while loop
	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	<-- 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);
```

```java
boolean invalidInput = true;
int i;
Scanner keyboard = new Scanner(System.in);
do {
    System.out.print("Type a number between 1 and 10: ");
    i = keyboard.nextInt();
    if (i >= 1 && i <= 10){
        invalidInput = false;
    } else {
        System.out.println("Error. Try again!");
    }
} while(invalidInput == true);

System.out.println("Success: " + i + " is a valid number");

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 boleano "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  }
```

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

```
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.

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

i>=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, está operación se ejecutará, reduciendo el valor de "i" en "2".

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 intensión, el código y el programa que estamos evaluando.

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

```java
//i is never updated, so this loops infinitely
while(i < 10){
  System.out.println(i);
}

// wrong condition i<0, causing an infinite loop
for (int i = 10; i>0; i++){
  System.out.println(i);
}
```