

# Usando un bucle for ¿Cuándo? ¿Cómo?

Mostrar por pantalla la tabla de multiplicar del 4:

Repetir 10 veces la instrucción `System.out.println("4 x " + i + " = " + 4 * i);` con  $1 \leq i \leq 10$

```
BlueJ: BlueJ: Ventana de Terminal...
Tabla del 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40

System.out.println("Tabla del 4");
System.out.println("4 x 1 = " + 4 * 1);
System.out.println("4 x 2 = " + 4 * 2);
System.out.println("4 x 3 = " + 4 * 3);
System.out.println("4 x 4 = " + 4 * 4);
System.out.println("4 x 5 = " + 4 * 5);
System.out.println("4 x 6 = " + 4 * 6);
System.out.println("4 x 7 = " + 4 * 7);
System.out.println("4 x 8 = " + 4 * 8);
System.out.println("4 x 9 = " + 4 * 9);
System.out.println("4 x 10 = " + 4 * 10);
```

**Escribe** un programa Java `TablaDel4` que, usando un bucle for, muestre por pantalla la tabla del 4, tal como aparece en el Terminal de BlueJ de la figura anterior

# Usando un bucle for

## ¿Cuándo? ¿Cómo? Mecánica del for

**Solución del puzle del Ejemplo 1 del tema (Sesión 1):** diseña un método que devuelva el producto de a y b, enteros no negativos, **SIN** usar el operador \*

En TestSProductoSinUsarX  
de ejemplos S1 – Tema 6

```
// PRECONDICIÓN: a >= 0 AND b >= 0
public static int productoSinUsarX(int a, int b) {
    int i = 0; int res = 0;
    while (i != a) {
        res = res + b;
        i++;
    }
    return res;
}
```

**Completa** el cuerpo del siguiente método usando el bucle for **equivalente** al while que aparece en **productoSinUsarX**

```
// PRECONDICIÓN: a >= 0 AND b >= 0
public static int productoSinUsarXFor(int a, int b) {

}
}
```

# Usando un bucle for

## ¿Cuándo? ¿Cómo? Mecánica del for

**Solución del puzle del Ejemplo 2 del tema (Sesión 1):** diseña un método que devuelva la suma de las cifras de *a*, entero no negativo, **SIN** usar `Math.log10`

En TestSumarCifras de  
ejemplos S1 – Tema 6

```
// PRECONDICIÓN: a >= 0
public static int sumarCifras(int a) {
    int res = 0; int i = a;
    while (i != 0) { // i > 0
        res = res + i % 10;
        i = i / 10;
    }
    return res;
}
```

**Completa** el cuerpo del siguiente método usando el bucle for **equivalente** al `while` que aparece en `sumarCifras`

```
// PRECONDICIÓN: a >= 0
public static int sumarCifrasFor(int a) {

}

}
```

# Usando un bucle for

## ¿Cuándo? ¿Cómo? Mecánica del for: Ejercicios propuestos

- **Nº 8 Transparencias:** ¿qué muestra por pantalla el siguiente bucle?

```
for (int cuenta = 1; cuenta < 5; cuenta++) {  
    System.out.println(2 * cuenta);  
}
```

- **Nº 9 Transparencias:** ¿qué muestra por pantalla el siguiente bucle?

```
for (int i = 0, j = 10; i < j; i++, j--) {  
    System.out.println("i: " + i + " j: " + j);  
}
```

- **Ejercicios 3 y 4 del capítulo 8 del libro de la asignatura**
- **Cuestión:** en el programa Ejercicio2Capitulo8Libro del proyecto BlueJ *sesion1 de ejercicios – Tema 6* tienes 2 versiones de sumarHasta, un método que devuelve la suma de los n primeros números naturales ¿Cuál de los bucles while que aparecen en ellas traducirías a bucle for? ¿Por qué?

# Usando un bucle for

## Bucles Anidados: las tablas de multiplicar

El programa TablaDe14 del proyecto BlueJ *ejercicios S4 – Tema 6* muestra por pantalla la tabla de multiplicar del 4

```
System.out.println("Tabla del 4");  
  
for (int i = 1; i <= 10; i++) {  
    System.out.println("4 x " + i + " = " + 4 * i);  
}
```

¿Cómo lo modificarías para que mostrara, una tras otra, las tablas de multiplicar del 1 al 10?

# Usando un bucle for

## Bucles Anidados: trazas

- ¿Qué muestra por pantalla el siguiente código?

```
int nfil = 4, ncol = 3;
for (int i = 1; i <= nfil; i++) {
    for (int j = 1; j <= ncol; j++) {
        System.out.print(i + "-" + j + " ");
    }
    System.out.println();
}
```

- ¿Y este?

```
for (int i = 0; i <= n; i++) {
    for (int j = 1; j <= i; j++) { System.out.print('Z'); }
    for (int j = 1; j <= n; j++) { System.out.print('A'); }
    for (int j = 1; j <= n - i; j++) { System.out.print('Z'); }
    System.out.println();
}
```

Ejecuta los métodos `traza1` y `traza2` de la clase `Trazas` (del package **sesion4** de *ejercicios – Tema 6*) y comprueba tus respuestas; para una mejor comprensión de cómo funcionan, puede resultar útil poner puntos de ruptura en las líneas de cada uno de los `for`

# Usando un bucle for

## Bucles Anidados: dibujar un rectángulo

Dibuja en pantalla un rectángulo, usando asteriscos ('\*')

base	altura	dibujo
3	2	*** ***
6	3	***** ***** *****
4	5	***** ***** ***** ***** *****

**Algoritmo:**

**Código:**

```
// PRECONDICIÓN
```

# Usando un bucle for

## Bucles Anidados: dibujar un triángulo

Dibuja en pantalla un triángulo rectángulo **isósceles** de altura dada, con '\*'

altura	dibujo
2	* **
3	* ** ***
4	* ** *** ****

**Algoritmo:**

**Código:**

```
// PRECONDICIÓN
```



# Usando un bucle for

## Bucles Anidados: dibujar un cuadrado

Dibuja en pantalla un cuadrado de lado  $n$ , con '\*' y blancos (' '), como el que muestra la siguiente figura para  $n = 5$

```
* * * * *  
*   *   *  
*   *   *  
*   *   *  
* * * * *
```

**Algoritmo:**

**Código:**

```
// PRECONDICIÓN:
```

## Usando un bucle for

### Bucles Anidados: dibujar una figura de n líneas

Dibuja en pantalla una figura con  $n$  líneas tal que: cada línea tendrá  $n - 1$  asteriscos y el carácter 'a' en la posición diagonal principal de la figura. Por ejemplo, para  $n = 5$  se mostraría por pantalla lo siguiente:

a\*\*\*\* Algoritmo:

\*a\*\*\*

\*\*a\*\*

\*\*\*a\*

\*\*\*\*a

Código:

```
// PRECONDICIÓN:
```

# Usando un bucle for

## Bucles Anidados: dibujar un paralelogramo

Dibuja en pantalla un paralelogramo de altura y base dadas. Por ejemplo, para altura = 4 y base = 9 mostraría por pantalla lo siguiente:


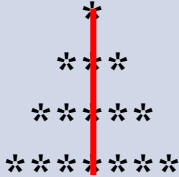
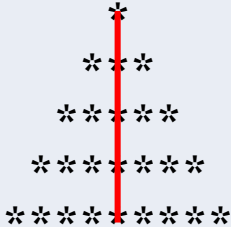
```
*****  
*****  
*****  
*****
```

**Algoritmo:**

# Usando un bucle for

## Bucles Anidados: dibujar la copa de un árbol

Dibuja en pantalla un triángulo isósceles de base dada, con '\*' y blancos (' ')

base	dibujo
3	 <pre>  *  * *</pre>
7	 <pre>    *   * *  * * * * * * *</pre>
9	 <pre>    *   * *  * * * * * * * * * * * *</pre>

**Algoritmo:**