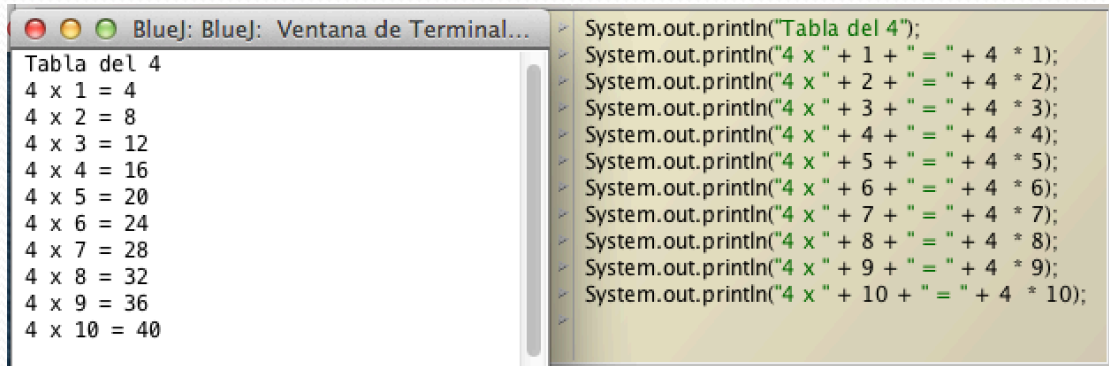


La necesidad de repetir instrucciones y de la instrucción “repetir”

Observa el resultado que aparece en el *Terminal de BlueJ* cuando se ejecuta el código del *Code Pad* para los ejemplos que siguen. Luego, responde (por escrito) las cuestiones que se plantean para cada ejemplo

1. Mostrar por pantalla la tabla de multiplicar del 4



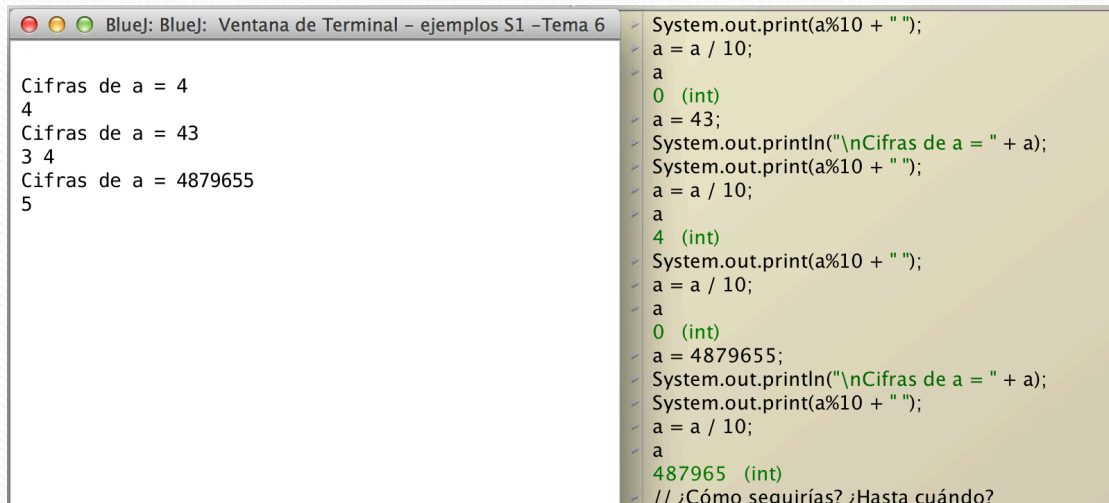
```
BlueJ: BlueJ: Ventana de Terminal...
Tabla del 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40

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);
```

Cuestiones:

- ¿Qué instrucción se repite?
- ¿Cuántas veces?
- ¿Sería fácil escribir el código para mostrar la tabla del 4 hasta el 4 x 200? ¿Y la del 896?

2. Mostrar por pantalla las cifras de a, un int positivo dado



```
BlueJ: BlueJ: Ventana de Terminal - ejemplos S1 -Tema 6
Cifras de a = 4
4
Cifras de a = 43
3 4
Cifras de a = 4879655
5

System.out.print(a%10 + " ");
a = a / 10;
a
0 (int)
a = 43;
System.out.println("\nCifras de a = " + a);
System.out.print(a%10 + " ");
a = a / 10;
a
4 (int)
System.out.print(a%10 + " ");
a = a / 10;
a
0 (int)
a = 4879655;
System.out.println("\nCifras de a = " + a);
System.out.print(a%10 + " ");
a = a / 10;
a
487965 (int)
// ¿Cómo seguirías? ¿Hasta cuándo?
```

Para cada valor de a, ...

- ¿Qué instrucciones se repiten?
- ¿Cuántas veces?
- ¿Variaría alguna de tus respuestas anteriores si a se leyera de teclado?

¿Cómo aprender a partir de ejemplos la instrucción a repetir?

(a) ¿Cuál es el (valor del) siguiente término? ...

Observa la solución para la primera serie, por si te ayuda a resolver las siguientes

1, 1, 2, 6, 24, ... **120**

1, 2, 4, 8, 16, 32, ...

0, 1, 3, 6, 10, 15, ...

(b) ¿Qué relación general existe entre un término n y el anterior a él?

Observa la solución para la primera serie, por si te ayuda a resolver las siguientes

1, 1, 2, 6, 24, ... **120**

1, 2, 4, 8, 16, 32, ...

0, 1, 3, 6, 10, 15, ...

si $n > 0$, $n * (n - 1)!$; si $n = 0$, $(0)! = 1$

(c) ¿De qué función matemática hablamos?

Observa la solución para la primera serie, por si te ayuda a resolver las siguientes

1, 1, 2, 6, 24, ... **120**

1, 2, 4, 8, 16, 32, ...

0, 1, 3, 6, 10, 15, ...

$n!$ (factorial de n)

¿Qué estrategia o tipo de razonamiento has usado para descubrirlo?

¿Se parece a la que empleaste para resolver el puzle de Multiplicación “a la Rusa”

La iteración como estrategia de diseño - Ejemplo 1

Diseña un método que devuelva el producto de a y b, enteros no negativos, **SIN** usar el operador *

// **PRECONDICIÓN:** $a \geq 0$ AND $b \geq 0$

```
public static int productoSinUsarX(int a, int b) {
```

```
    int res = ; int i = ;
```

```
    while (  ) {
```

```
        
```

```
    }
```

```
    // PARADA: tras repetición...
```

```
    return res;
```

```
}
```

// **INICIO**, o repetición 0

// **!PARADA** == GUARDA

// **CUERPO:** instrucción a repetir

Estrategia:

La iteración como estrategia de diseño - Ejemplo 2

Diseña un método que devuelva la suma de las cifras de a, entero no negativo, **SIN** usar `Math.log10`

// **PRECONDICIÓN:** $a \geq 0$

```
public static int sumarCifras(int a) {
```

```
    int res = ; int i = ;
```

```
    while (  ) {
```

```
        
```

```
    }
```

```
    // PARADA: tras repetición...
```

```
    return res;
```

```
}
```

// **INICIO**, o repetición 0

// **!PARADA** == GUARDA

// **CUERPO:** instrucción a repetir

Estrategia:

La iteración como estrategia de diseño - Mecánica del while



BlueJ: ejemplos S1 -Tema 6

- **Sitúa**, como en la imagen, un punto de ruptura en la línea 32 del main de TestSumarCifras

Luego, **traza** la ejecución del método `sumarCifras` con ayuda del depurador de BlueJ y **observa** cómo se modifican las variables `a` (contador) y `res` en cada repetición (*Step*), hasta que termine su ejecución. Usa **435**, por ejemplo, como valor de `a`

- **Repite** el proceso anterior usando como argumento un valor negativo, por ejemplo **-435** ... **¿Qué sucede?**

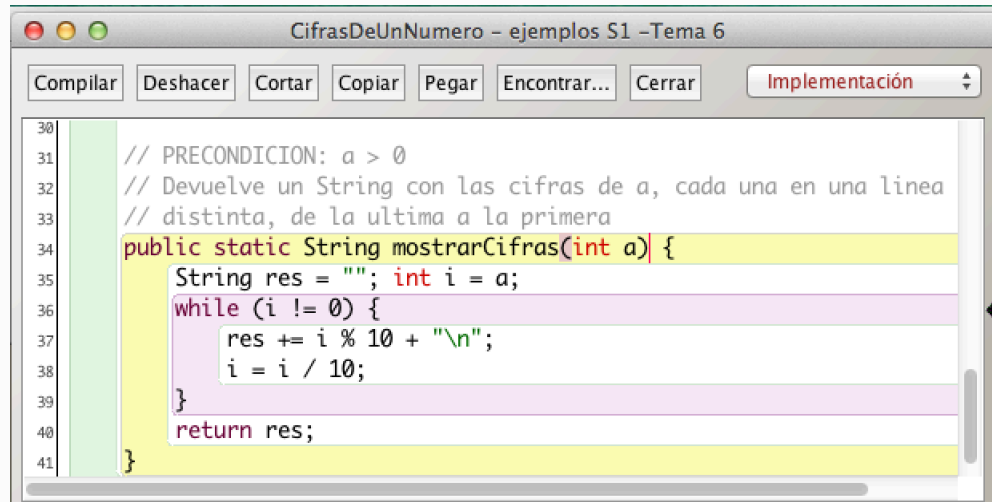
La iteración como estrategia de diseño

Cuestión sobre el Ejemplo 2



BlueJ: ejemplos S1 -Tema 6

- **Edita** el programa CifrasDeUnNumero del proyecto y **observa** el código de su método mostrarCifras, que devuelve el nº de cifras de a, un nº entero positivo



```
30
31 // PRECONDICION: a > 0
32 // Devuelve un String con las cifras de a, cada una en una linea
33 // distinta, de la ultima a la primera
34 public static String mostrarCifras(int a) {
35     String res = ""; int i = a;
36     while (i != 0) {
37         res += i % 10 + "\n";
38         i = i / 10;
39     }
40     return res;
41 }
```

- **Modificando** donde creas necesario la estrategia del ejemplo 2, **escribe** la estrategia seguida para obtener el bucle de mostrarCifras

Luego, en base a la estrategia que has escrito, **indica** por qué el cuerpo, inicio y guarda del bucle de mostrarCifras solo pueden ser los que son

La iteración como estrategia de diseño

(Examen PoliformaT) Mecánica y semántica del while: ejercicio 2

Sea n una variable `int` inicializada a un valor no negativo. Sin usar el método `Math.pow`, completa el siguiente bucle para que muestre por pantalla los n primeros **cuadrados perfectos**, de 1 en adelante y en líneas separadas. Por ejemplo, si $n = 4$, se escribirán en líneas separadas 1, 4, 9 y 16

```
int i =  ;  
while (  ) {  
    System.out.println(  );  
    i++;  
}  

```

// INICIO, o repetición 0

// CUERPO: instrucción a repetir

// PARADA: tras repetición...

Estrategia:

La iteración como estrategia de diseño

(Examen PoliformaT) Mecánica y semántica del while: ejercicio 3

Sea n una variable `int` inicializada a un valor no negativo. Sin usar el método `Math.pow`, completa el siguiente bucle para que muestre por pantalla los **primeros cuadrados perfectos menores o iguales que n** , de 1 en adelante y en líneas separadas. Por ejemplo, si $n = 5$, se escribirán en líneas separadas **1** y **4**

```
int i =  ;  
while (  ) {  
    System.out.println();  
    i++;  
}  

```

// INICIO, o repetición 0

// !PARADA == GUARDA

// CUERPO: instrucción a repetir

// PARADA: tras repetición...

Estrategia:

La iteración como estrategia de diseño

(Examen PoliformaT) Mecánica y semántica del while: ejercicio 4

Sea n una variable `int` inicializada a un valor no negativo. Completa el bucle del siguiente método para que calcule el factorial de n tal y como muestra su traza para $n = 5$

// PRECONDICIÓN: $n \geq 0$

```
public static int factorial(int a) {
```

```
    int i = ; int fact = ; // INICIO
```

```
    while (  ) {
```

```
        
```

```
        // CUERPO
```

```
    }
```

```
    // PARADA:
```

```
    return fact;
```

```
}
```

i	fact
0	1
1	1
2	2
3	6
4	24
5	120

¿Estrategia?

La iteración como estrategia de diseño - Ejercicio nº 2 Capítulo 8 del libro

Diseña un método que devuelva la suma de los n primeros números naturales

// **PRECONDICIÓN:** $n \geq 0$

```
public static int sumarHasta(int n) {
```

Estrategia: i es el último nº sumado

```
    int res = ; int i = ;
```

// INICIO

```
    while (  ) {
```

```
        
```

// CUERPO

```
    }  
    // PARADA:
```

```
    return res;
```

```
}
```

```
public static int sumarHasta(int n) {
```

Estrategia: i es el siguiente nº a sumar

```
    int res = ; int i = ;
```

// INICIO

```
    while (  ) {
```

```
        
```

// CUERPO

```
    }  
    // PARADA:
```

```
    return res;
```

```
}
```