SESIÓN 3: Estructuras de control: Repetitivas, Iterativas o Bucles

Objetivos

- Saber construir y utilizar una estructura repetitiva **while, for, do-while**.
- Saber elegir el tipo de bucle adecuado para un problema determinado.
- Ser capaces de construir bucles anidados.
- Evitar la construcción de bucles infinitos.

Nota importante: Siga el esquema de nombrado de paquetes que se indicó en la sesión 01 es decir: **org.ip.sesion03**. En ese paquete se crearán todos los programas que se proponen en la sesión dándoles un nombre al programa y que se indica en cada ejercicio **entre paréntesis y en negrita**.
Al final de la sesión, el alumno deberá cargar el trabajo realizado a su repositorio indicando la clave correspondiente a la sesión.

Ejercicios propuestos

1. Desarrolle un programa que lea desde teclado un número indeterminado de enteros (separados por espacios en blanco) y muestre cuantos son positivos, cuantos negativos, número total de valores leídos, suma de los valores positivos, suma de los valores negativos, media de todos los valores leídos (no se contabiliza el 0 para la media). El programa termina cuando la entrada es *0*, denominado *valor centinela*. Prueba con una estructura *while* y con *for*. (**BucleCentinela**).

   Nota: Uso del Scanner en Java

```
import java.util.Scanner;
class EjemploLectura {
  public static void main(String arg[]){
    String variableString;
    // Se declara e inicializa una instancia  de la clase Scanner
    Scanner entrada=new Scanner(System.in);
    System.out.print("Ingrese un texto: ");
    variableString = entrada.next();
    // Para leer valores enteros, reales, etc. desde teclado, hay que declarar
    // las variables apropiadas y luego llamar a nextInt(), nextFloat(), etc.
    System.out.println("Texto introducido: " + variableString);
  }
}
```

   Ejemplo de ejecución:

```
Introduce valores enteros, el programa termina si la entrada es 0
1 2 -1 3 0
El número de positivos es 3
El número de negativos es 1
El número total de valores leidos es 4
La suma de positivos es  6
La suma de negativos es  -1
La media de los valores es  1.25
```

2. Desarrolle un programa que permita obtener el **MCD** (máximo común divisor) de dos enteros positivos haciendo uso del algoritmo de Euclides. **(Euclides)**. Para ello se divide el *dividendo* (x) entre el *divisor* (y) y se obtiene un *cociente* (q_1) y un *resto* (r_1) si dicho resto es distinto de 0 se divide nuevamente el anterior *divisor* entre el *resto* obtenido. Se continua el proceso hasta obtener un resto 0, el último divisor es el MCD.

$x \mid \underline{y}$ $r_1 \quad q_1$	$y \mid \underline{r_1}$ $r_2 \quad q_2$	$r_1 \mid \underline{r_2}$ $r_3 \quad q_3$	....	$r_{n-2} \mid \underline{r_{n-1}}$ $r_n \quad q_n$
$r_1 \neq 0$	$r_2 \neq 0$	$r_3 \neq 0$	....	$r_n = 0 \Rightarrow r_{n-1}$ es el mcd

Ejemplo de ejecución:

```
Introduce el primer valor entero positivo
-3
Introduce el primer valor entero positivo
25
Introduce el segundo valor entero positivo
5
El MCD de 25 y 5 es 5
```

3. Desarrolle un programa que muestre los K primeros términos de la serie de Fibonacci.

$f_0 = 0$

$f_1 = 1$

$f_i = f_{i-1} + f_{i-2}$ para $i \geq 2$ **(Fibonacci)**

Ejemplo: $f_0 = 0; f_1 = 1, f_2 = 1, f_3 = 2, f_4 = 3, f_5 = 5, f_6 = 8, f_7 = 13, f_8 = 21, f_9 = 34, f_{10} = 55, \ldots$

Ejemplo de ejecución:

```
¿Hasta qué término de la serie de Fibonacci quieres mostrar?
10
0    1    1    2    3    5    8    13    21    34    55
```

4. Un entero positivo es un *Número Perfecto* si es igual a la suma de todos sus divisores positivos excluyendo el mismo. Por ejemplo, 6 es el primer número perfecto porque sus divisores (excluyendo el mismo) son 1, 2 y 3 cuya suma 1 + 2 + 3 = 6 coincide con el propio número. Desarrolle un programa que pida un número y muestre por consola si es perfecto. **(NumeroPerfecto)**

Ejemplo de ejecución:

```
Introduzca un número entero positivo para saber si es perfecto
28
El número es perfecto
```

Trabajo autónomo

5. Desarrolle un programa que muestre si un número entero positivo (>1) es primo. En el archivo EjerciciosResueltos_Tema01.pdf, visto en clase de teoría. En el ejemplo número 12 muestra dos maneras de hacerlo y deja planteada una tercera. Resuelve esta última. Prueba la ejecución para números primos y no primos. **(Primo)**

Ejemplo de ejecución:

```
Introduce un número (>1) para saber si primo
0
Introduce un número (>1) para saber si primo
9
NO ES PRIMO
```

6. Dado un número natural no superior a 10, implemente un programa que muestre por consola un triángulo rectángulo con lado igual al número dado tal y como se muestra en la siguiente figura. **(TrianguloNumeros)**

Ejemplo de ejecución:

```
Triangulo rectangulo de numeros para un valor de lado = 10
                    0
                  1 0
                2 1 0
              3 2 1 0
            4 3 2 1 0
          5 4 3 2 1 0
        6 5 4 3 2 1 0
      7 6 5 4 3 2 1 0
    8 7 6 5 4 3 2 1 0
9 8 7 6 5 4 3 2 1 0
```

7. Dados un número mayor que cero, *lado*. Desarrolle un programa en Java que muestre por consola un cuadrado de asteriscos tal y como se indica en la siguiente figura. **(CuadradoAsteriscos)**

Ejemplo de ejecución:

```
Cuadrado de asteriscos de lado 7

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

8. Dado un número mayor que cero, *lado*, tal que *3 < lado < 15*. Desarrolle un programa en Java que muestre por consola un triángulo rectángulo de asteriscos tal y como se indica en la siguiente figura. **(TrianguloAsteriscos)**

Ejemplo de ejecución:

```
Triangulo rectangulo de asteriscos de lado 7

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

9. El número **PI** (π) puede ser calculado mediante la siguiente serie:

$$\pi(i) = 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \cdots + \frac{(-1)^{i+1}}{2i-1}\right)$$

Implementar programa Java (**EstimarPI**) que genere dicha serie para un *i* dado (ese *i* corresponde con la precisión del cálculo de la estimación de π, es decir, cuanto mayor sea el valor de *i* más precisa será la estimación de π) y muestre el último valor generado según la serie mostrada en consola. La generación de la serie debe ser la misma que se muestra como salida a continuación y se ha de comprobar con el valor de π que tiene en Math (**Math.PI**).

Ejemplo de ejecución:

```
i                    PI(i)
---------------------------
1                4,00000000
101              3,15149340
201              3,14656775
301              3,14491490
401              3,14408642
501              3,14358866
601              3,14325655
701              3,14301919
801              3,14284109
901              3,14270253

Precision =      1000
miPI =           3,142702531161429
Math.PI =        3,141592653589793
```