

# **UD2.- Elements Auxiliars.**

## Elements auxiliars als bucles (variables)

Els elements auxiliars son **variables que realitzen funcions especifices dins d'un programa**, i per la seua gran utilitat, freqüència d'ús i peculiaritats, convé fer un estudi separat d'aquestes.

Quan s'utilitzen bucles dins d'un programa, ens pots enfrontar a dues possibles situacions:

- Que coneguem a priori quantes vegades han de repetir-se les instruccions (**repetició definida**),
- Que el número de vegades que s'hagen de repetir les instruccions depenga d'un valor que es no coneix fins al moment de l'execució del bucle (**repetició indefinida**).

En el primer cas (repeticions definides) necessitarem una variable que actue com un **comptador**, en la qual es registre el número d'iteracions que es vagen executant.

En el segon cas (repeticions indefinides) generalment es controlen mitjançant **interruptors** o **banderes**, o bé, amb valors **sentinella**.

Estes variables funcionen exactament igual als tres tipus de bucles vistos anteriorment **FOR, WHILE I DO-WHILE**, recordem la seua sintaxi:

```
for (inicialització ; condició ; increment)
{
    // bloc d'accions (Instruccions);
}
```

```
while (condició)
{
    // bloc d'accions (Instruccions);
}
```

```
do {
    // bloc d'accions (Instruccions);
} while (condició)
```

# 1.- Comptadors

Anomenem **comptador** a una **variable** que **s'incrementa en una quantitat constant** quan repetim una acció un número determinat de vegades, es a dir, a cada iteració d'un bucle.

Tot comptador **s'ha d'inicialitzar amb un valor inicial** (0, 1...)

Sobre una **variable comptador** es realitzen dues operacions bàsiques: **inicialització** i **increment o decrement**, segons siga el cas.

```
comptador = valor_inicial;           // comptador = 0;
```

Cada vegada que aparega l'esdeveniment (operació) a comptar s'ha **d'incrementar** o **decrementar** en una **quantitat fixa**.

```
comptador = comptador + Increment; // comptador = comptador + 2;
comptador = comptador - Decrement; // comptador = comptador - 5;
```

Els comptadors més utilitzats tenen increments o decrements d'un en un (i++,i--).

Els **comptador** s'utilitzen en els següents casos:

- Per a comptabilitzar el **número de voltes que és necessari repetir una acció** (variable de control d'un bucle).
- Per a **comptar un succés particular** sol·licitat per l'enunciat del problema.

**Un comptador ha d'inicialitzar-se a un valor inicial (normalment a zero) i incrementar-se cada vegada que ocorregui un succés.**

**Exemple: Programa que compta i imprimeix els números parells entre l'1 i el 200, amb un comptador que suma de dos en dos.**

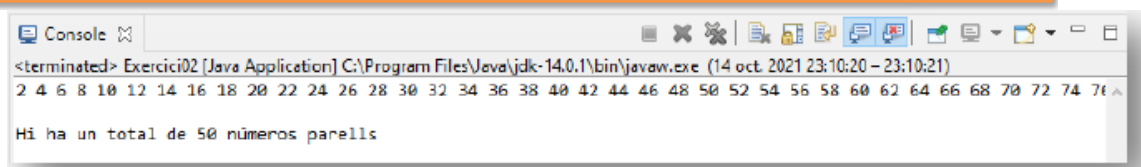
```
public static void main(String[] args) {

    int controlaBucle, comptaParells;

    controlaBucle=2;
    comptaParells=0;

    while(controlaBucle<=100) {
        System.out.print(controlaBucle + " ");
        controlaBucle=controlaBucle+2;
        comptaParells=comptaParells+1;
    }

    System.out.println("\n\nHi ha un total de " + comptaParells + " números parells");
}
```



**Exemple: Programa que demana un número per teclat, compta i imprimeix tots els múltiples de 3 des de l'unitat fins al número introduït.**

```
public static void main(String[] args) {
    int compta, numMax, i;

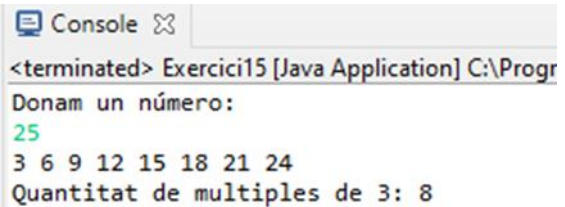
    Scanner datoEntrada = new Scanner(System.in);
    System.out.println("Donam un número: ");

    numMax=datoEntrada.nextInt();
    compta=0;

    for (i=3;i<=numMax;i=i+3) {
        compta++;
        System.out.print(i + " ");
    }

    System.out.print("\n");
    System.out.println("Quantitat de multiples de 3: " + compta);
}
```

En este exemple tenim **dos comptadors** la **variable i** que ens permet controlar les iteracions del bucle for i la **variable compta** que s'incrementa en una unitat a cada iteració del bucle.



```
<terminated> Exercici15 [Java Application] C:\Progr
Donam un número:
25
3 6 9 12 15 18 21 24
Quantitat de multiples de 3: 8
```

**NOTA:** Podem tindre més d'un comptador en la declaració del bucle for, i incrementar/decrementar el dos a l'hora, el següent exemple és equivalent a l'anterior.

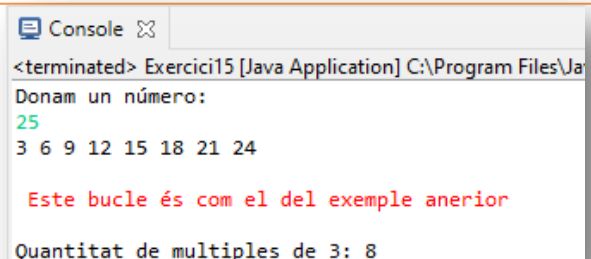
```
public static void main(String[] args) {
    int numMax, compta, i;
    Scanner datoEntrada = new Scanner(System.in);

    System.out.println("Donam un número: ");

    numMax=datoEntrada.nextInt();

    for (i=3, compta=0; i<=numMax; i=i+3, compta++) {
        System.out.print(i + " ");
    }

    System.err.print("\n\n Este bucle és com el del exemple anterior \n\n ");
    System.out.println("Quantitat de múltiples
}
```



```
<terminated> Exercici15 [Java Application] C:\Program Files\Ja
Donam un número:
25
3 6 9 12 15 18 21 24

Este bucle és com el del exemple anerior

Quantitat de multiples de 3: 8
```

## ERRORS COMUNS

**NO podem declarar comptadors de distints tipus en la capçalera del bucle FOR**, per exemple la següent declaració és incorrecta:

```
for (float i=3, int compta=0; i<=numMax; i=i+3, compta++)
```

En canvi, si declarem les variables abans del bucle, JAVA sí que ens ho permet:

```
int i;
float compta;

for (i=3, compta=0; i<=numMax; i=i+3, compta++)
```

Recorda que si declarem una variable dins de la capçalera d'un bucle, esta sols existeix dins del mateix bucle, i s'allibera quan eixim del bucle. **(Àmbit de les variables)**

```
public static void main(String[] args) {

    int numMax;

    Scanner datoEntrada = new Scanner(System.in);
    System.out.println("Donam un número: ");

    numMax=datoEntrada.nextInt();

    for (int i=3, compta=0; i<=numMax; i=i+3, compta++) {

        System.out.print(i + " ");

    }

    System.err.print("\n\n Este bucle és com el del exemple anterior \n\n ");
    System.out.println("Quantitat de multiples de 3: " + compta);

}
```

A l'anterior exemple la variable compta sols existeix en el bloc d'instruccions del bucle FOR (**variable local**), una volta fora del bucle, ja s'ha alliberat, i per tant no existeix i es produeix un error de compilació.

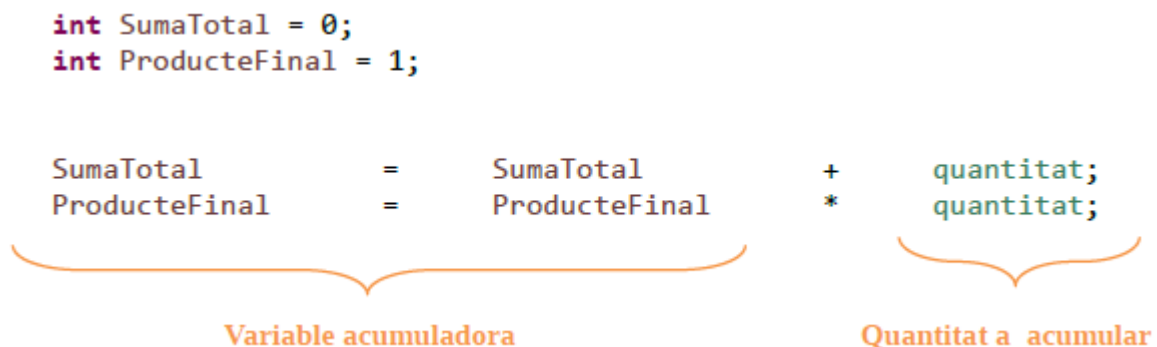
## 2.- Acumuladors

Un **acumulador** és una **variable** que té com a objectiu **acumular quantitats successives** obtingudes en realitzar la mateixa operació, es a dir, a cada iteració del bucle.

L'ús més habitual d'un acumulador és obtenir **sumes i productes**. Igual que amb els comptadors, per a poder utilitzar un acumulador cal realitzar sobre ells dues operacions bàsiques: **inicialització i increment**.

En el cas d'obtindre sumes, l'acumulador s'inicialitza en zero i en el cas dels productes en un, per tal de no afectar el resultat.

Una volta obtingut i emmagatzemat en una variable la quantitat a acumular l'afegim a la variable acumuladora:



**EXEMPLE:** Programa que calcula i escriu la suma i el producte dels 10 primers números naturals:

```

public static void main(String[] args) {

    int i, suma, producte;

    suma = 0;
    producte = 1;

    for (i = 1; i <= 10; i++) {

        suma = suma + i;
        producte = producte * i;
    }

    System.out.println("La suma és " + suma + " y el producte és " + producte);
}

```

Console

```

<terminated> Ejercicio09 [Java Application] C:\Program Files\
La suma és 55 y el producte és 3628800

```

En este exemple tenim dos acumuladors la variable **suma** que ens permet acumular la suma a cada iteració del bucle i la variable **producte** que acumula el producte a cada iteració del bucle.

La variable **i** actua com a comptador del bucle i com a **quantitat** acumulada a cada iteració.

**EXEMPLE : Programa que llegueix per teclat un número positiu N i calcule el seu factorial N!**

El factorial d'un número es defineix com el producte de tots els números anteriors o iguals a ells. per exemple, factorial de 5 ->  $5! = 5*4*3*2*1 = 120$

```
public static void main(String[] args) {  
  
    Scanner lligNumero=new Scanner(System.in);  
    //Creació de l'objecte "lligNumero" per poder utilitzar els mètodes de la classe Scanner  
  
    int numero, factorial;  
        // utilitzem la variable numero com a COMPTADOR del bucle  
        // utilitzem la variable factorial com ACUMULADOR del bucle  
  
    System.out.print("Introdueix el número per calcular el seu factorial: ");  
    numero = lligNumero.nextInt();  
  
    factorial = 1;  
  
    while (numero > 1) {  
        factorial = factorial * numero;  
        // Per cada iteració del bucle, multipliquem el comptador per el número acumulat  
  
        numero--; //equivalent a numero=numero-1  
    }  
  
    System.out.println("El seu factorial és " + factorial);  
}
```

### 3.- Interruptors o Banderes

Un **interruptor** o **bandera** és una **variable** que pot prendre **dos** possibles **valors** al llarg de l'execució del programa, estos són: 1 (encesa/oberta) i 0 (apagat/tancat), d'ahí el seu nom.

Normalment són de tipus **enter (0,1)** o **booleanes (true, false)**

La seua funció principal és que **unes certes instruccions s'executen mentre tinga un valor determinat**.

**EXEMPLE: Programa que llig de teclat una seqüència de notes (entre 0 i 10), la seqüència acaba al introduir el valor -1 i ens diu si hi ha alguna nota amb valor 10**

```
public static void main(String[] args) { Scanner lligNota = new Scanner(System.in);

float nota;
boolean hay10 = false; //Variable de CONTROL inicialment a FALSE

System.out.print("Introdueix una nota entre 0 y 10 (-1 per acabar):");
nota = lligNota.nextFloat();

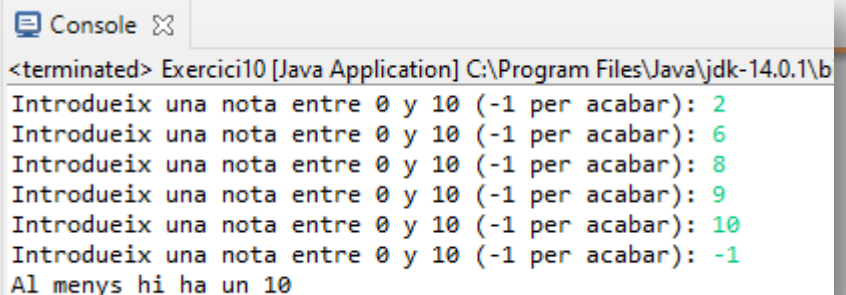
//Com utilitzem un bucle while, hem de llegir una nota abans d'entrar al bucle, si la el valor introduït és
distint de -1 entrarà dins del bucle

while (nota != -1) {
    if (nota == 10) {
        hay10 = true; //En cas d'haver un 10, posem la variable booleana a valor TRUE
    }

    System.out.print("Introdueix una nota entre 0 y 10 (-1 per acabar):"); nota = lligNota.nextFloat();
}

if (hay10==true) //Si hem introduït algun 10 mostrarem per pantalla un missatge o altre

    System.out.println("Al menys hi ha un 10");
else
    System.out.println("No hi ha notes de 10");
}
```



```
<terminated> Ejercicio10 [Java Application] C:\Program Files\Java\jdk-14.0.1\bin
Introdueix una nota entre 0 y 10 (-1 per acabar): 2
Introdueix una nota entre 0 y 10 (-1 per acabar): 6
Introdueix una nota entre 0 y 10 (-1 per acabar): 8
Introdueix una nota entre 0 y 10 (-1 per acabar): 9
Introdueix una nota entre 0 y 10 (-1 per acabar): 10
Introdueix una nota entre 0 y 10 (-1 per acabar): -1
Al menys hi ha un 10
```

En este exemple tenim dos variables de control, la variable **nota** que a banda de guardar les notes introduïdes per l'usuari controla quan acaba el bucle (valor -1).

La variable **hay10** ens avisarà si s'ha introduït alguna nota amb valor 10.