# Programació

## UT 2.3 Programació estructurada. Estructures de repetició

# Estructures de repetició

- Permeten repetir una mateixa seqüència d'instruccions múltiples vegades, mentre es complisca certa condició
- **bucle**: conjunt d'instruccions que s'han de de repetir un cert nombre de vegades
- **iteració**: cada execució individual del bucle

# Control d'estructures iteratives

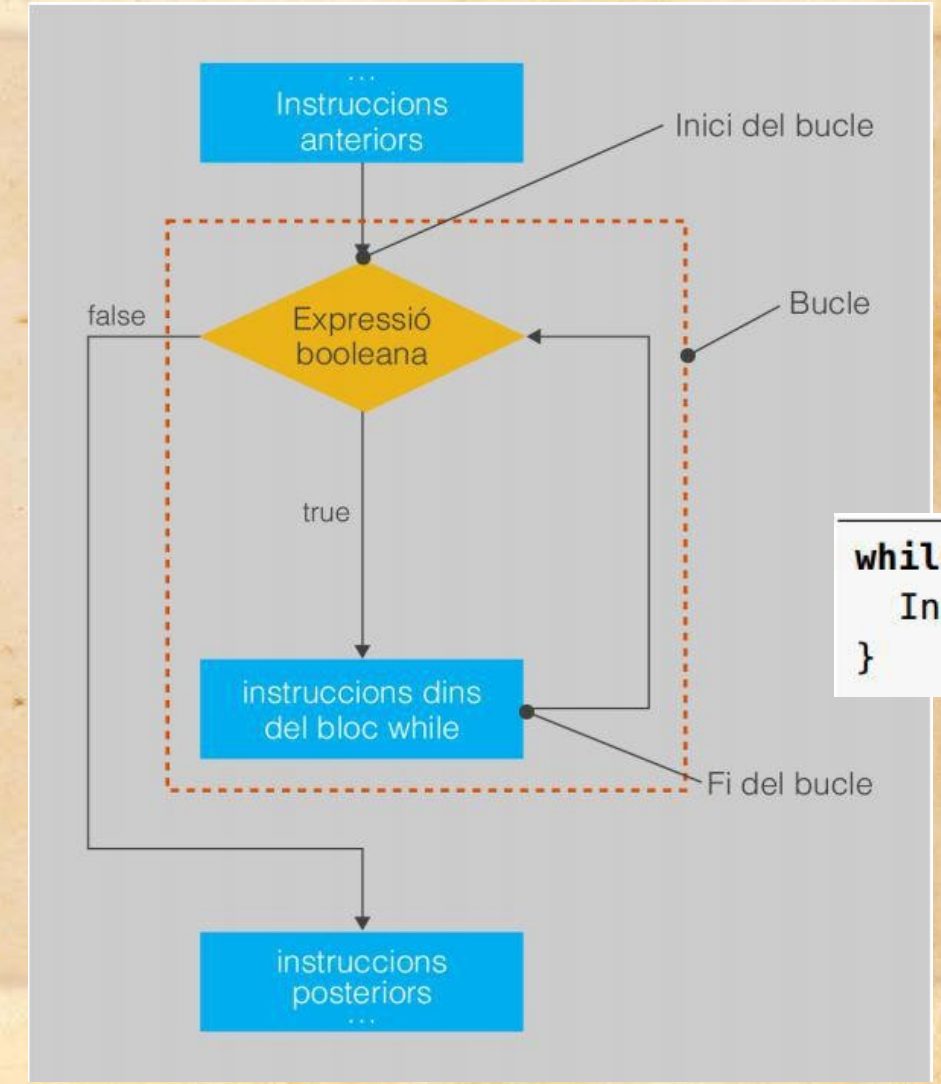
- Cal sempre assegurar-se que el bucle acabarà, en cas contrari, serà un **bucle infinit**
- Per **evitar este problema**, es fan ús de **variables de control**

  - **Comptador**: una variable que incrementa o decrementa a cada iteració el mateix valor

  - **Acumulador**: una variable que augmenta o disminueix a cada iteració un valor igual o diferent

  - **Semàfor**: una variable (normalment booleana) que canvia el seu valor per permetre o prohibir l'execució d'iteracions

# Estructura iterativa (while)




Si l'expressió booleana no es compleix, **no s'executarà mai** el bloc d'instruccions definit.

```
while (expressió booleana) {
  Instruccions per executar dins del bucle
}
```

# Pràctica 8

- Escriu 1 programa que dibuixe una línia horitzontal creada a base del símbol '_' i que ocupe 100 posicions

*RECORDA: És important que realitzes aquestes pràctiques, faces proves i anotes qualsevol problema per a preguntar al professor.*

# A tindre a compte

- Els identificadors dels *comptadors* **solen** ser sempre i, j, k …. (i successius).
- Els *comptadors* **solen** començar sempre pel valor 0 en comptes del valor 1 (això no vol dir que sempre tinga que ser així, però **millora la llegibilitat** ja que segueix l'estàndard)

# Pràctica 9

- **Modifica el codi font del programa de la pràctica 8 per a que s'adapte a les recomanacions descrites anteriorment.**
- **A més, modifica el programa per a que demane el nombre de barres baixes per teclat en a lloc de definir-ho com a constant.**
- **No oblides verificar el tipus de dada de l'entrada. Si no introdueixen un nombre enter, acaba el programa.**

# Pràctica 10

- **Realitza un programa que mostre la taula de multiplicar de un nombre que s'introduirà per teclat, respectant les recomanacions indicades anteriorment.**

  Igual que abans, si no s'introdueix un valor enter, acaba el programa.

```
Introdueix el nombre del qual vols mostrar la taula: hola
No has introduït un valor enter.
```

```
Introdueix el nombre del qual vols mostrar la taula: 6
6 x 0 = 0
6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
6 x 4 = 24
6 x 5 = 30
6 x 6 = 36
6 x 7 = 42
6 x 8 = 48
6 x 9 = 54
6 x 10 = 60
```

# Pràctica 11

- **Crea un programa que permeta sumar tots els valors enters múltiples de 3, dins d'un interval entre 0 i un valor introduït per teclat.**
  - **Resol este programa de 2 maneres:**
    - **Fent ús de l'operador %**
    - **Sense fer ús de l'operador %**

- Comprova les dades d'entrada. Si no són correctes, acaba.

# Pràctica 12

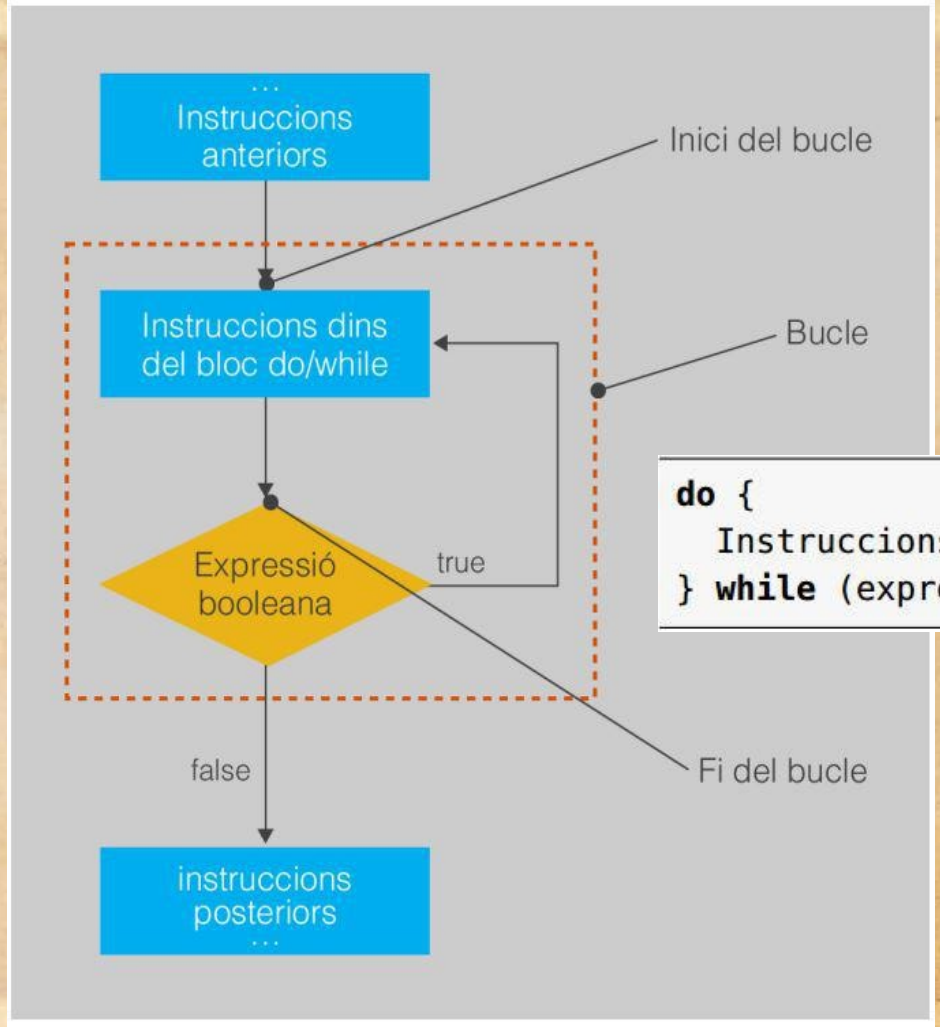
- **Realitza un programa que, donat dos nombres enters, calcule el residu de dividir un per l'altre, sense fer ús de l'operador %**
- Comprova les dades d'entrada. Si no són correctes, acaba el programa.

# Pràctica 13

- **Realitza un millor control de les dades introduïdes per teclat, en este cas, en comptes d'acabar el programa, ha de tornar a preguntar per esta dada si s'introdueix erròniament. Fes-ho en un programa a part, on únicament demane un enter i realitze esta revisió fins que la dada s'introduisca correctament.**

# Estructura iterativa (do-while)

Permet repetir l'execució mentre es verifique la condició lògica. Com a mínim **s'executarà una volta** el bloc d'instruccions definit.




```
do {
  Instruccions per executar dins del bucle
} while (expressió booleana);
```

# Pràctica 14

- **Realitza un programa que verifique l'entrada per teclat de un nombre enter entre 0 i 10.**
- **El programa ha de demanar la dada en cas de que l'entrada siga errònia (ja ho vas fer amb while). Utilitza ara una estructura iterativa "do - while " per fer aquesta revisió.**

# Pràctica 15

- **Aplica la mateixa manera de demanar una dada correcta als programes creats anteriorment i que siga oportú fer-ho. Per exemple, al demanar un preu.**

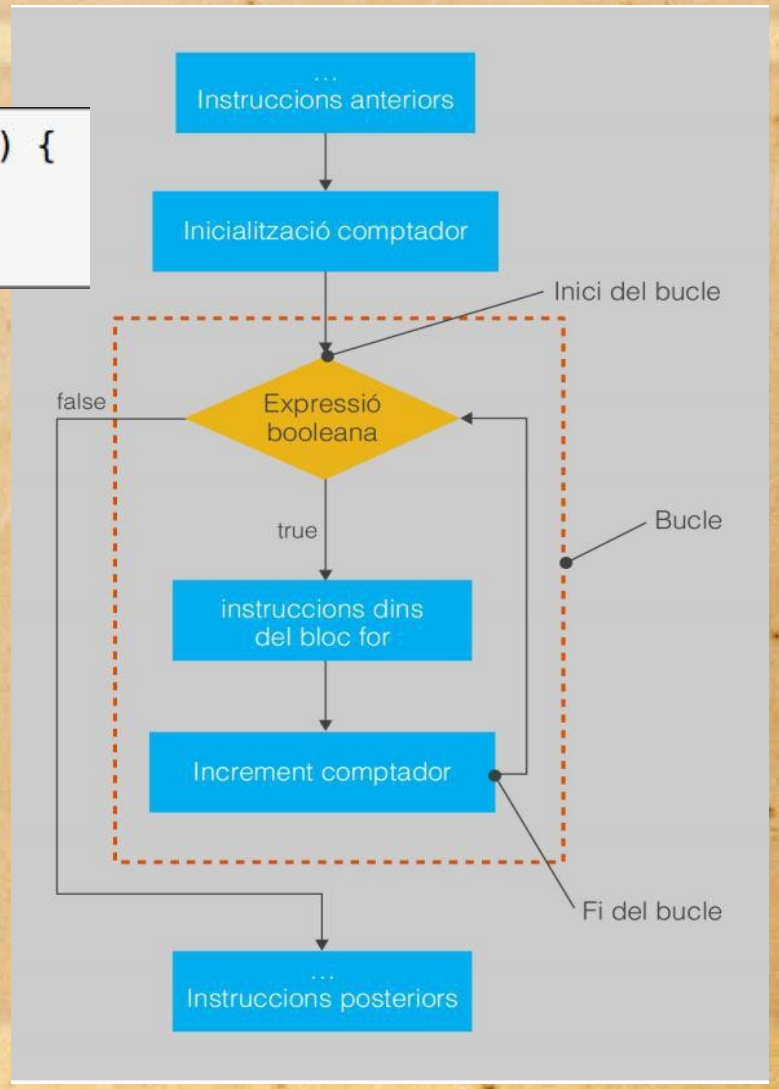
# Estructura iterativa (for)

```
for (inicialització comptador ; expressió booleana ; increment comptador) {
  Instruccions per executar dins del bucle
}
```

Permet repetir un nombre determinat de voltes un conjunt d'instruccions

**exemple:**

```
for (int i = 0; i < 20; i ++) {
   ... ..
}
```



# Parts del bucle *for*

- **Inicialització de comptador**: inicialitza una variable de tipus enter que servirà com a comptador.
- **Expressió booleana**: la condició lógica. Si es certa es torna a repetir el cos del bucle (torna a iterar)
- **Increment**: instrucció que modifica el comptador. Esta instrucció s'executa al final de cada iteració. L'increment pot ser també negatiu (decrement).

# A tindre en compte

- **Tota sentència "for" té el seu equivalent** mitjançant 1 sentència "while".
- L'increment/decrement del comptador utilitza (normalment) l'operador unari (++) o (--). El resultat d'esta operació és la mateixa que incrementar/decrementar en una unitat el comptador donat. **Això no lleva que es puga utilitzar una altra manera d'incrementar un comptador .**

# Pràctica 16

- **Adapta el programa que mostrava la taula de multiplicar, aquesta vegada, fent ús de la sentència "for".**

# Pràctica 17

- **Adapta el programa de la suma de múltiples de 3, fent ús d'un bucle "for".**

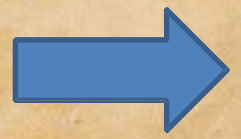
# Pràctica 18

- **Modifica la pràctica 16. Aquesta volta, es preguntarà a l'usuari el número màxim de la taula de multiplicar a mostrar.**

# Traça de variables

En programació, realitzar la traça d'una variable es refereix a realitzar un seguiment des de la seua declaració fins a la seua eliminació de memoria, dels valors que va adquirint a cada moment.

```java
public class Bucles {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        int a = 100;
        int total = 0;

        while (a < 120) {

            if (a % 3 == 0) {
                a = a + 2;
                total = total + a;


            }
            else {
                a++;
                total = total - a;
            }

            System.out.println("Total:" + total);
        }
    }
}
```



```
Total:-101
Total:-203
Total:-99
Total:-204
Total:-97
Total:-205
Total:-95
Total:-206
Total:-93
Total:-207
Total:-91
Total:-208
Total:-89
Total:-209
```

*A partir d'ara quan utilitzes estructures repetitives i no obtingues el resultat que esperaves, hauràs de crear traces de variables per a observar el funcionament pas per pas del programa. Més endavant parlarem de la depuració.*