Nom: ..... Data: ..... Curs/Grup: .....

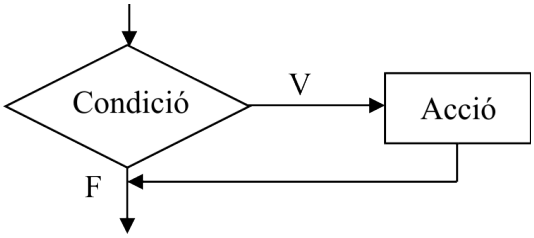
Programació i Tractament de dades
Java. Estructures condicionals

Fitxa J.05

Recordem quines eren i com les implementam en Java.

Condicional simple

Representa l'execució d'una acció només en el cas que es compleixi una determinada condició.



si <condició> **llavors**
    accions;
fsi;

```
if (expressió booleana) {
    // Accions a executar
}
```

Consideracions:

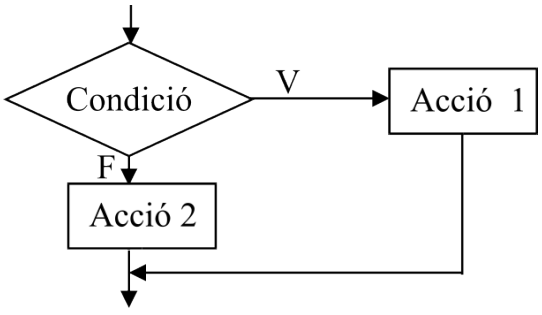
- L'expressió booleana ha d'anar sempre entre parèntesis.
- Les instruccions a executar van entre claus.
- La línia on acaba el bloc no du punt i coma.
- Per comoditat es solen sagnar les instruccions del bloc.
- Si sols hi ha una instrucció les claus són opcionals.

Exemple:

```
if (e<0){
    System.out.println("El nombre és negatiu");
}
```

Condicional alternatiu

Representa l'execució d'una acció en el cas que es compleixi una determinada condició i d'una altra en el cas que no es compleixi.



si <condició> **llavors**
    acció_1;
sinó
    acció_2;
fsi;

```
if (expressió booleana) {
    // Accions a executar si la condició és certa
}
else {
    // Accions a executar si la condició és falsa
}
```

Exemple:

```
if (e<0){
    System.out.println("El nombre és negatiu");
} else {
    System.out.println("El nombre és positiu");
}
```

Condicional múltiple

En aquest cas en van encadenant blocs if-else.

```
if (expressió booleana 1) {
     // Accions si exp1 és certa;
} else if (expressió booleana 2) {
     // Accions si exp2 és certa;
[...]
} else if (expressió booleana n) {
     // Accions si exp-n és certa;
} else {
     // Accions si no s'ha satisfet cap expressió anterior;
}
```

Amb aquesta estructura aconseguim només executar un bloc, malgrat hi hagi diverses expressions que pugin avaluar "cert".

El bloc "else" és opcional.

Condicional selectiu

En aquest condicional s'avalua una expressió que ha de ser entera o caràcter.

cas <selector> **fer**
valor1: acció_1;
valor2: acció_2;
...
valorn: acció_n;
altrs: acció;
fcas;

```
switch (selector) {
     case valor1:
          // accions_1
          break; // Opcional
     case valor2:
          // accions_2
          break; // Opcional
     ...
     case valorn:
          // accions_n
          break; // Opcional
     default:
          // acció
}
```

La sentencia "break" serveix per sortir de l'estructura del "switch". Si no el posam s'aniran executant la resta de casos, fins que trobi un "break".

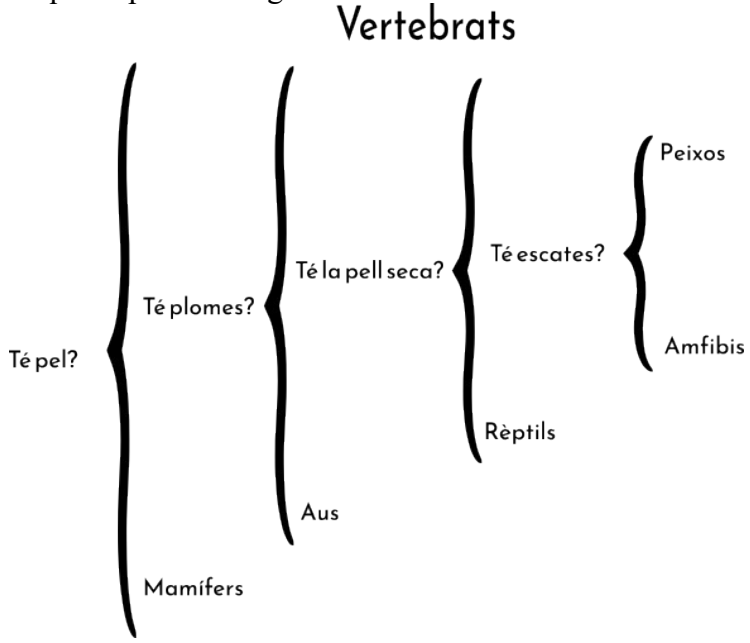
Exemple:

```
int dia=8;
switch (dia){
     case 1: System.out.println("Dilluns");
          break;
     case 2: System.out.println("Dimarts)";
          break;
     ...
     case 7: System.out.println(="Diumenge");
          break;
     default: diaSetmana="Dia incorrecte";
}
```

Ex. 1. Fes un programa que llegeixi un número enter i digui si és parell o senar

Ex. 2. Fes un programa que llegeixi tres números per teclat i ens digui si estan ordenats o no.

Ex. 3. Fes un programa que implenti la següent clau dicotòmica de vertebrats:



Ex. 4. Fes un programa que llegeixi un any per teclat i retorni si és de traspàs o no.

Ex. 5. Un operari cobra segons les hores treballades i els metres de cable instal·lats.
Definirem el preu de les hores (30€) i els metres de cable (0,50€) com a constants.
El programa es ha de demanar:
- les hores treballades
- el nombre de metres instal·lats

Ha de retornar
- "El preu brut és..."
- "El preu amb IVA és ..."

Ex. 6. En una empresa necessiten un programa per calcular el salari dels treballadors. Els paguen setmanalment. Han d'introduir:
- Nombre d'hores treballades la setmana
- Preu per hora del treballador

Es considera que un treballador comença a fer hores extres a partir de la 35a hora. Les hores extres es paguen un 50% més que les hores normals.
Ha de retornar l'import final a pagar.