


ABANS DE PASSAR A FER PRÀCTIQUES...

ACABEM LA PART DE TEORIA

Taules com a paràmetres de funcions

- Quan haguem de fer funcions que treballin amb taules, les haurem de passar per paràmetre
- Haurem de passar per paràmetre: la **taula** i el **nombre d'elements** que té



```
funció mitjana (v: taula[] de real, n: enter) retorna real és  
...
```

ffunció

algorisme ...

```
dades: taula [MAX] de real;
```

```
dades2: taula [MAX] de real;
```

```
... $ MAX és la mida màxima de les taules 'dades' i 'dades2', que  
$ contenen 'n_dades1' i 'n_dades2' elements respectivament
```

```
mitj := mitjana (dades, n_dades1);
```


```
mitj2 := mitjana (dades2, n_dades2);
```

Exemple. Dues taules (vectors) de reals, cadascuna amb un nombre diferent d'elements.

Taules com a paràmetres de funcions

Matrius com a paràmetres

- Per passar taules de més d'una dimensió per paràmetre, es fa igual que amb 1 dimensió, senzillament necessitem passar la mida de totes les dimensions.



```
funció comptar_zeros (v: taula[][] d'enter, n_f: enter, n_c:enter)
retorna enter és

...
ffunció
algorisme
var
dades: taula [N_FIL][N_COL] d'enter;
fvar

...
n_zeros := comptar_zeros (dades, N_FIL, N_COL);
```

Pas de paràmetres

Modificació dels valors dels paràmetres

- En alguns casos el procediment ha de variar (canviar) els valors dels paràmetres.
- Exemple: funció que intercanvia dos valors.

```
acció intercanvia ( a: enter, b: enter ) és  
var aux: enter; fvar  
inici  
    aux := a;  
    a := b;  
    b := aux;  
facció  
...  
algorisme ...  
var x, y: enter; fvar  
...  
    intercanvia (x, y);  
...
```

En aquest cas, dins el procediment es modifica el valor de 'a' i de 'b'.

Pas de paràmetres

Pas per valor vs. pas per referència

- Distinció **molt important**
- En programació, distingim entre dos mecanismes de pas de paràmetres:

Pas per valor

Paràmetres **no** modificables
Qualsevol canvi que fem dins el procediment no tindrà efecte quan sortim del procediment

```
x := 1;  
y := 2;  
...  
intercanvia (x, y);  
...  
$ x val 1  
$ y val 2
```

Pas per referència

Paràmetres modificables
Els canvis que fem dins el procediment es mantindran al sortir-ne

```
x := 1;  
y := 2;  
...  
intercanvia (x, y);  
...  
$ x val 2  
$ y val 1
```

pass by reference



fillCup()

pass by value



fillCup()

www.penjee.com

Pas de paràmetres

Pas per valor vs. pas per referència

- Distinció **molt important**
- En programació, distingim entre dos mecanismes de pas de paràmetres:

Pas per valor

Paràmetres **no** modificables

Qualsevol canvi que fem dins el procediment no tindrà efecte quan sortim del procediment

```
x := 1;  
y := 2;  
...  
intercanvia (x, y);  
...  
$ x val 1  
$ y val 2
```

Pas per referència

procediment es mantindran al
sortir-ne

```
x := 1;  
y := 2;  
...  
$ x val 2  
$ y val 1
```

Ho aprendrem a fer quan vegem punters

Cada llenguatge de programació gestiona el pas de paràmetres de manera diferent

En llenguatge C...

... es fa servir pas per valor

*Això vol dir que mai podrem
modificar els paràmetres?*

No! Es fan servir adreces de memòria (anomenats punters o apuntadors) i llavors es pot accedir al contingut i modificar-lo

Pas per valor

Exemple de pas per valor. Què retorna aquest programa?

```
funció calcula (a: enter, b: enter, c:  
enter) retorna enter és  
var  
    suma : enter;  
fvar  
inici  
    c := a + b;  
    suma := c * 100;  
    retorna (suma);  
ffunció
```

Procediment "calcula"

```
x = 2  
y = 3  
z = 7  
k = 500
```

```
algorisme principal és  
var  
    x, y, z, k: enter;  
fvar  
inici  
    x := 2;  
    y := 3;  
    z := 7;  
    k := calcula(x,y,z);  
    escriure("x = ", x);  
    escriure("y = ", y);  
    escriure("z = ", z);  
    escriure("k = ", k);  
falgorisme
```

Programa principal