TEORIA

PROGRAMACIÓ CIENTÍFICA

T2

Tipus de dades

(part 1: variables i constants)

PSEUDOCODI

Què és el pseudocodi?

- El pseudocodi és una **descripció** en **llenguatge planer** dels passos d'un algorisme o d'un sistema.

- El pseudocodi fa servir **convencions d'estructura** com fan servir els llenguatges de programació, però es pretén que sigui un llenguatge que els humans puguin llegir, no que les màquines puguin interpretar.
- **Omet els detalls tècnics** que són essencials perquè la màquina entengui l'algorisme, i no inclou les particularitats de cap llenguatge en concret.

- **L'objectiu** d'usar pseudocodi és:
  - Escriure un codi que tots els humans entenguin
  - Escriure un codi que sigui independent del llenguatge de programació que usarem per implementar-lo
- Important: Ens serveix per **separar la fase de disseny d'un algorisme de la fase d'implementació**

```
algorisme
  var
    edat : enter;
  fvar
   escriure("Quants anys tens?");
   llegir(edat);
   si edat < 20 llavors
     escriure("Ets de la generació Z!");
   fsi
falgorisme
```

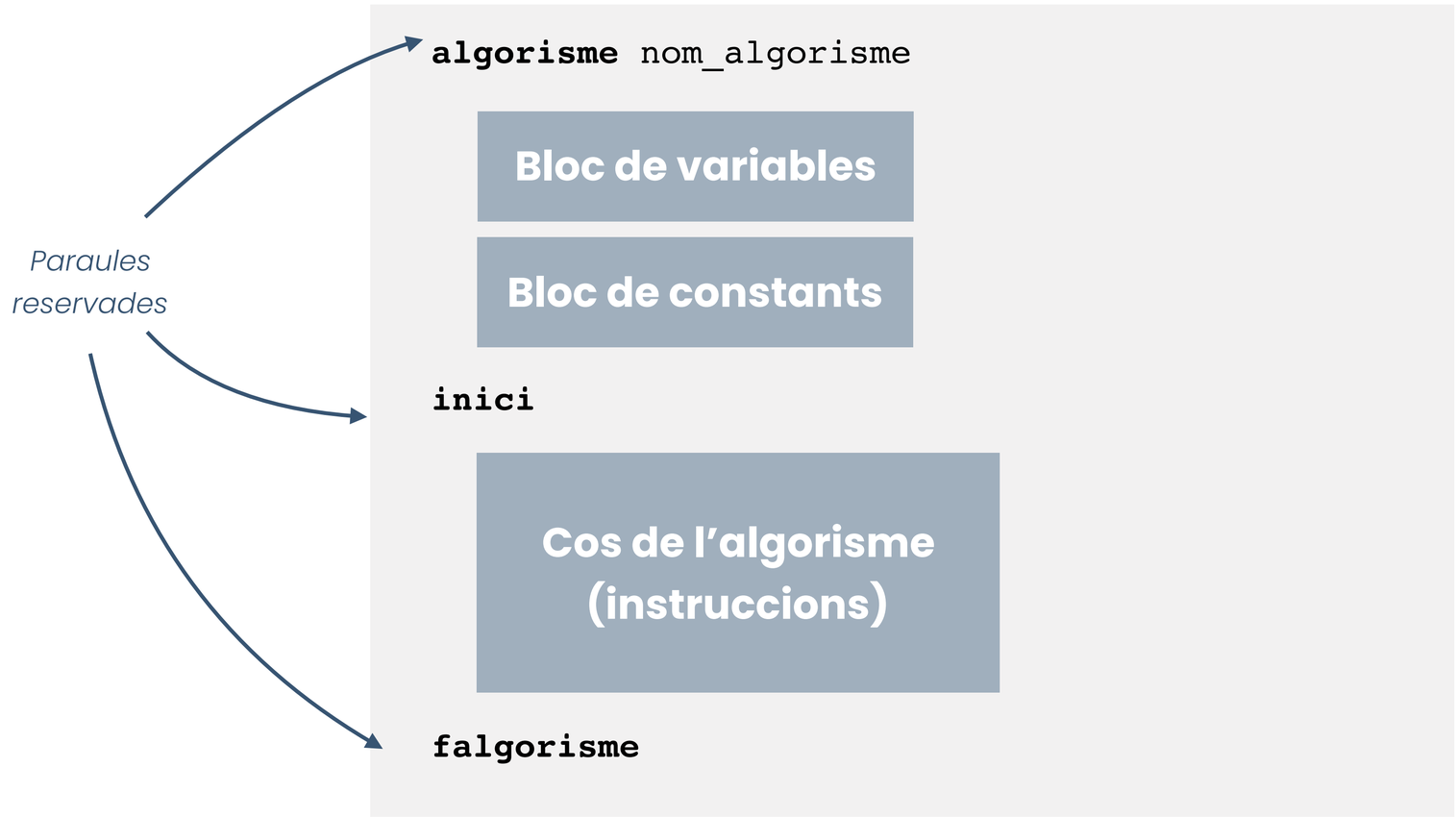
Pseudocodi d'un algorisme tontet

```c
#include <stdio.h>

int main(){
  int edat;
  printf("Quants anys tens?\n");
  scanf("%d",&edat);
  if (edat < 20){
      printf("Ets de la generacio Z!\n");
  }
  return 0;
}
```

Codi equivalent en C

Estructura del pseudocodi

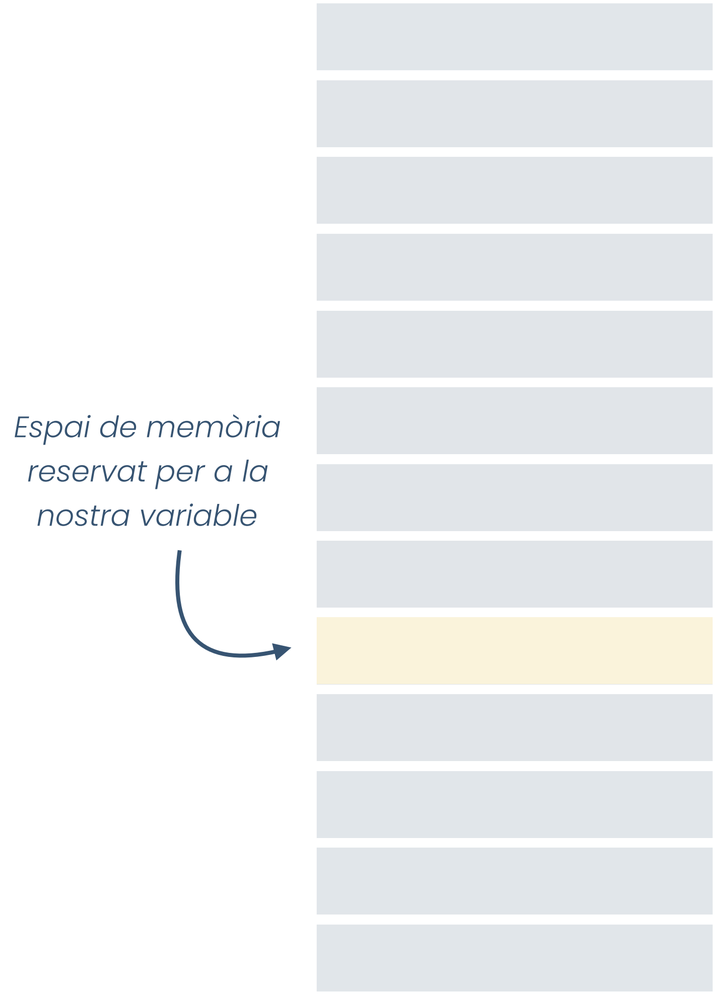


VARIABLES I CONSTANTS

Variables i constants

Declaració de variables

- Una variable emmagatzema a la memòria un valor que es pot consultar i modificar dins un programa
- Com declarem una variable?
  - Assignar **l'identificador** que farem servir quan vulguem treballar amb la variable i indicar-ne el **tipus**
  - Com a resultat, es reserva un espai de memòria de la mida del tipus.



- En pseudocodi:

```
var
    identificador: tipus;
    identificador2: tipus;
fvar
```

Definim totes les variables dins una zona delimitada per `var` i `fvar`

Utilitzem el punt-i-coma en acabar una instrucció, per analogia a la majoria de llenguatges de programació.

Variables i constants

Identificador d'una variable

- Cada llenguatge de programació té les seves pròpies regles de quins caràcters pot contenir una variable i quins no.

  Per exemple, en C una variable no pot començar amb un dígit

- A més, cada llenguatge té una guia d'estil amb recomanacions de com haurien d'estar escrites les variables

  Per exemple, en C es recomana que les variables no comencin amb underscore perquè es confondrien amb llibreries

  - Alguns estils
    - camelCase: `nomDeVariable`
    - PascalCase: `NomDeVariable`
    - snake_case: `nom_de_variable`
    - kebab-case: `nom-de-variable`

- Nosaltres:
  - Seqüència de lletres, números i guió baix (underscore _)
  - Snake case
  - El nom ha d'identificar el contingut:

```
nom_client
data_naixement
n_clients
```

Els identificadors no seran paraules "reservades" del pseudocodi (o del llenguatge de programació). Per exemple, `var` i `fvar` són paraules reservades del pseudocodi i no es poden usar per anomenar variables.

Les **convencions d'estil** són un conjunt de normes per un llenguatge de programació específic.

Recomanen estils de programació, bones pràctiques i mètodes per tal que l'aspecte del codi font sigui el mateix en tots els programes que usen aquell llenguatge, de manera que tothom el pugui llegir.

Aquestes recomanacions inclouen noms de variables, indentació, comentaris, declaracions, claus d'obertura i tancament... etc

Variables i constants

Constants

- Les constants són valors fixos que el programa no pot canviar durant la seva execució.
- El valor que prenen s'anomena **literal**. E.g. "hola", 35, ...
- Les constants poden ser de qualsevol tipus bàsic, com: enter, decimal, caràcter...
- Les constants es tracten com variables normals però els seus valors no poden canviar-se després de definir-se.

Per exemple, en C, si canvies el valor d'una constant, salta un error de compilació

- En pseudocodi:
  - Les definim entre les paraules reservades "const" i "fconst", abans de la definició de variables
  - Tot majúscules, per diferenciar-los de les variables

```
const
    IDENTIFICADOR := valor;
    PI := 3.1416;
    MAX_CLIENTS := 100;
fconst
```

*En C hi ha una altra manera de definir valors constants, la primitiva **#define**, que no genera variables.*
Ho veurem al laboratori

Hard-coding NO

```
...
a = b + 100;
c = 100 + a;
...
```

```
const
   MAX := 100;
fconst
...
a = b + MAX;
c = MAX + a;
...
```

Variables: nombres

Tipus de nombres

- Per treballar amb nombres cal analitzar si necessitem decimals o no.
  - Per treballar amb nombres enters, utilitzarem el tipus **enter**
  - Per treballar amb nombres amb decimals, utilitzarem el tipus **real**

- Per exemple, quin tipus de nombre faria servir per emmagatzemar...
  - El nombre total de participants d'un experiment?
  - El tant per cent d'homes i dones d'un experiment?

```
var
    total_participants: enter;
    percent_dones, percent_homes: real;
fvar
```

- Per assignar valors als nombres fem servir l'instrucció d'assignació
  - En pseucodi: **:=**

```
const
    PI:= 3,14159
fconst
var
    radi, perímetre: real;
fvar
inici
    radi := 5,0;
    perímetre := PI * radi * 2;
```

Parèntesi

Tipus de nombres

- A la pràctica, hi ha més tipus a més d'**enter** i **real:**

En pseudocodi	En C		
	Tipus	Mida	Domini o Rang
enter	short[1]	(2 bytes)16 bits	-32768 ... 32767
	int[1]	(4 bytes) 32 bits	-2147483648...2147483647
	long[1]	(8 bytes) 64 bits	-92233720368547758O8... 9223372036854775807
real	float	(4 bytes) 32 bits	-3.4E+38 ... 3.4E+38
	double	(8 bytes) 64 bits	-1.7E-308 ... 1.7E+308
	long double	(16 bytes) 128 bits	-3.4E-4092 ... 3.4E+4092

- Convé que ens parem a pensar quins possibles valors haurem de guardar en aquella variable i que triem la que sigui suficient per representar el valor

- Però quan escrivim pseudocodi, seguirem fent servir només **enter** i **real**.

Variables: nombres

Intercanvi de valors

- Com podem fer un algorisme senzill d'intercanvi de valors?
  - Tinc uns valors a les variables **a**, **b**, i vull que **b** contingui el valor de **a** i **a** contingui el valor de **b**

```
algorisme intercanvi
var
  a, b: enter;
fvar
inici
  a:= 2;
  b:= 5;
  $ Intercanviem
  b := a;
  a := b;
falgorisme
```

És correcte, aquest algorisme?

```
algorisme intercanvi
var
  a, b: enter;
  aux: enter;
fvar
inici
  a:= 2;
  b:= 5;
  $ Intercanviem
  aux := a;
  a := b;
  b := aux;
falgorisme
```

Variables: nombres

Intercanvi de valors

- Com podem fer un algorisme senzill d'intercanvi de valors?
  - Tinc uns valors a les variables **a**, **b**, i vull que **b** contingui el valor de **a** i **a** contingui el valor de **b**

```
algorisme intercanvi
var
  a, b: enter;
  aux: enter;
fvar
inici
  a:= 2;
  b:= 5;
  $ Intercanviem
  aux := a;
  a := b;
  b := aux;
falgorisme
```

Bonus: sense "aux"

```
algorisme intecanvi_v2
var
  a, b: enter;
fvar
inici
  a:= 2;
  b:= 5;
  $ Volem: a=5, b=2
  a := a + b; $ a=7
  b := a - b; $ b=2 OK!
  a := a - b; $ a=5 OK!
falgorisme
```

Variables: booleans i expressions lògiques

El tipus booleà

- El tipus booleà permet definir variables que desaran valors **cert** o **fals**.
  - Per exemple, poden desar el resultat d'una comparació

```
var
    a: enter;
    r1, r2: booleà;
fvar
inici
    a := 2;
    r1 := a = 2; $ r1 és cert
    r2 := a > 2;  $ r2 és fals
```

Variables: caràcters

El tipus caràcter

- Permet definir variables per emmagatzemar símbols que utilitzarem per representar informació textual
- Aquests símbols formen part d'un codi de caràcters (ASCII)
- Per definir un caràcter literal, el posem entre cometes simples: `'`
  - E.g. `'a', 'A', '1', '?' ...`
- Cada símbol té un número d'ordre dins el codi de caràcters
  - El número 'B' serà el número del símbol 'A' més 1, el número del 'C' serà el número d''A' més dos, etc. Les lletres minúscules tindran un número que estarà entre el número de la 'a' i el de la 'z', i les majúscules un que estarà entre el de la 'A' i el de la 'Z'.
  - Amb la diferència entre els números que ocupen una majúscula i la seva minúscula podem fer canvis de majúscula a minúscula, i viceversa. Si tenim un a variable entera `d` tal que `d:= 'A'-'a'` aleshores `'c' + d = 'C'`.

Alguns llenguatges de programació no permeten fer aquestes operacions. En aquests casos, el llenguatge ofereix utilitzar operadors de conversió de tipus. Per exemple, el procediment `ordinal('c')` podria convertir un caràcter a enter.

Com codifiquem caràcters no numèrics?

ASCII

- *Americal Standard Code for Information Interchange*
- Codificació de lletres fent servir 7 bits
- $2^7 = 128$ caràcters
  - [0 .. 31]: Caràcters de control ASCII
  - [32 .. 127]: ASCII printable characters

En tenim prou?

- **Extended ASCII:** 1 bit més = 8 bits
  - [0 .. 31]: Caràcters de control ASCII
  - [32 .. 127]: ASCII printable characters
  - [128 .. 255]: extended ASCII codes

En tenim prou???