UF1: INTRODUCCIÓ

Algorísmica

Mª Belén Tortosa Pedrón

1.- Algorísmica

1.1 Definicions

L'objectiu general d'un ordinador personal és el de proporcionar solucions a problemes quotidians d'una forma eficaç i senzilla. Per tal de poder-ho fer ens caldrà un mecanisme per a definir la resolució o el tractament d'un problema. A tal efecte usarem el que s'anomenen algorismes.....

Algorisme
c. 1275; del gentilici *al-khwarizmî*, sobrenom del matemàtic iranià arabitzat Muhammad ibn Musà (s.IX), natural de Coràsmia (*khwarizmî*), introductor de l'àlgebra a l'Europa medieval amb les seves traduccions] *m* LÒG /MAT Procediment de càlcul que consisteix a acomplir un seguit ordenat i finit d'instruccions que condueix, un cop especificades les dades, a la solució que el problema genèric en qüestió té per a les dades considerades.

1.1 Definicions

L'algorisme és la forma natural d'expressar la resolució d'un problema genèric de forma sistemàtica, i és per tant un pas previ imprescindible abans d'intentar elaborar un programa.

Important: Poden haver varis algorismes per a solucionar un mateix problema. De fet solen haver moltes maneres d'enfocar

Característiques:

- Ha de ser precís.
- Ha d'estar ben definit.
- Ha de consumir recursos finits
  - En temps
  - En memòria
  - En CPU.

L'algorisme és una abstracció, i per tant si està ben fet ha de ser GENERIC, és a dir, independent del llenguatge de programació amb el que posteriorment s'implementi

1.2 Cicle de desenvolupament del programari

El cicle complet de la programació consta doncs dels següents passos:

- Recollir informació sobre el problema i entendre la necessitat.
- Fase de definició o anàlisis:
    - o Analitzar possibles solucions i alternatives.
    - o Triar i definir un algorisme adequat
    - o Formalitzar l'algorisme amb alguna eina de representació adient
        (pseudocodi, fluxgrames, etc.)
- o Generar la documentació
- Fase de programació
    - o Programar l'algorisme.
    - o Fer proves unitàries
    - o Generar la documentació
- Provar el programa per part de l'interessat.
- Lliurar el programa, la documentació i el/s manual/s

1.3 Representació d'algorismes

Trobem quatre formes principals de suport a la representació

- **Organigrames o diagrames d'organització**

Serveixen per a identificar els personatges, les fonts de dades, i els programes que inter actuen en una situació determinada. S'identifiquen fluxes de dades.

- **Taules de casos o de decisió**

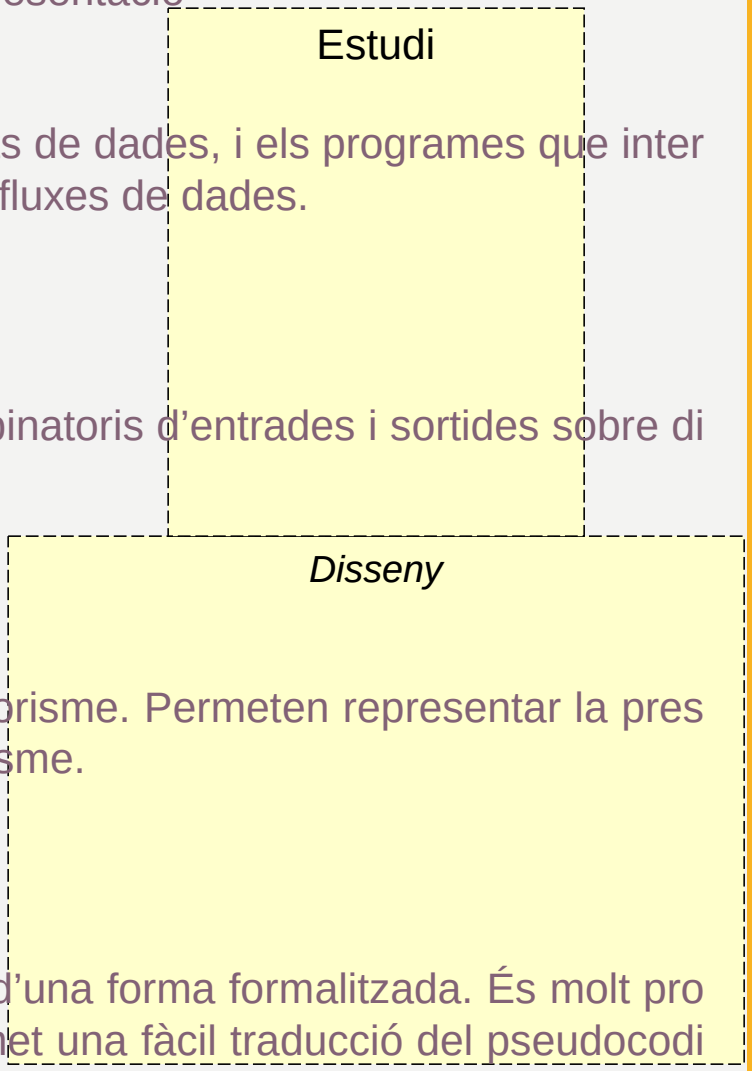
És una eina que permet presentar esquemes combinatoris d'entrades i sortides sobre di verses variables.

- **Diagrames de flux o ordinogrames**

S'utilitzen per a explicar el flux d'execució d'un algorisme. Permeten representar la pres a de decisions i la manipulació simbòlica de l'algorisme.

Pseudocodi

És la representació per escrit de l'algorisme, però d'una forma formalitzada. És molt pro per a la programació però encara és genèric. Permet una fàcil traducció del pseudocodi al llenguatge de programació triat.

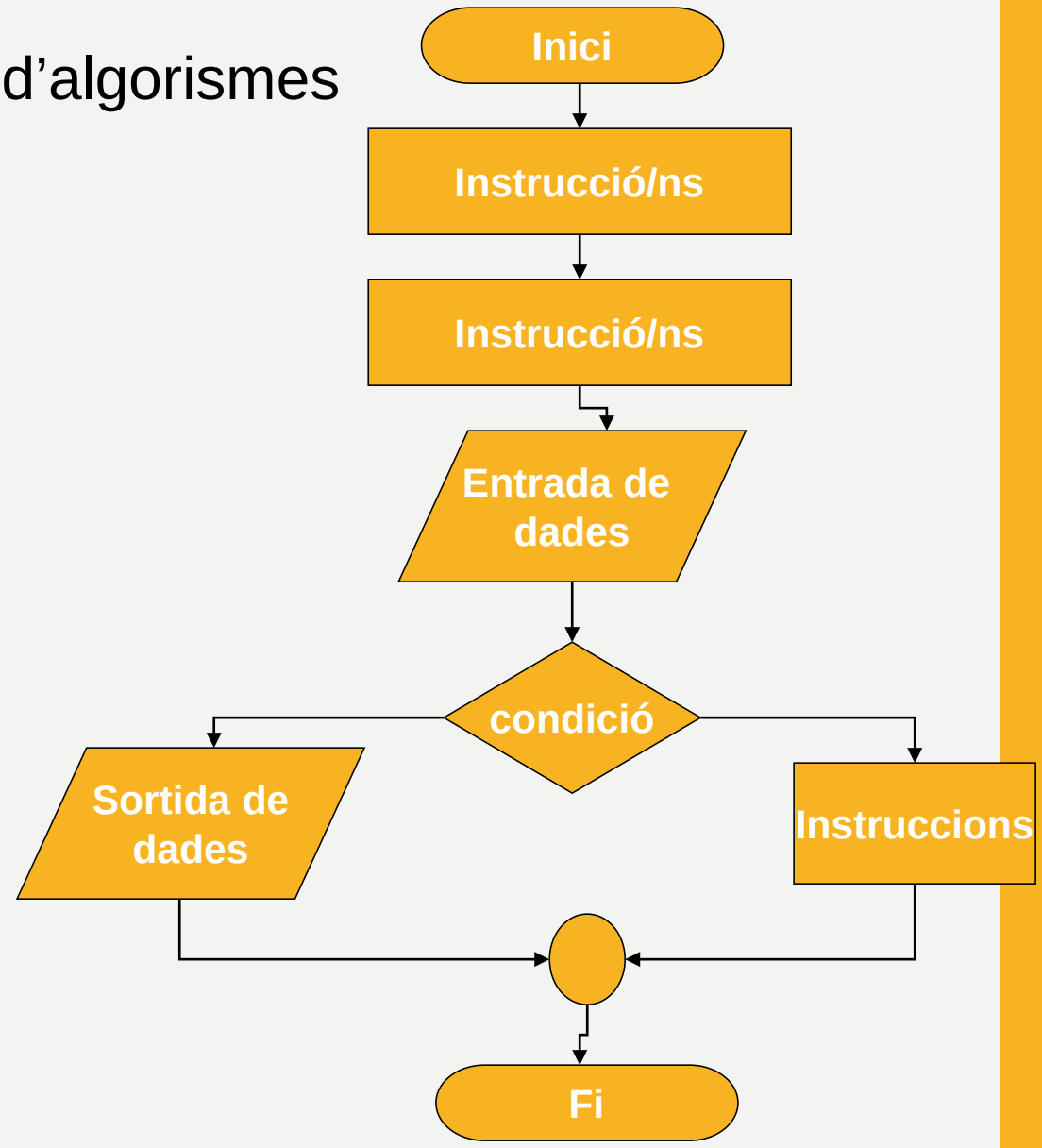


1.3 Representació d'algorismes

Diagrama de flux

Llista de Components bàsics

Línia de seqüència



1.3 Representació d'algorismes

b) Exercicis de diagrames de flux / quadres de decisió

A) Algorisme de multiplicació de dos núm eros

Problema: Calcular el producte de dos n úmeros

Dades d'entrada del problema:

- Els dos números

Sortida: el producte d'ambdós números

B) Algorisme de l'administratiu

Problema: Calcular quan s'ha de facturar a un client determinat.

Dades d'entrada del problema:

- Import total de factures del client
- % descompte aplicable al client

Sortida: Import final a pagar

1.3 Representació d'algorismes

b) Exercicis de diagrames de flux / quadres de decisió

C) Algorisme de valor absolut

Problema: Calcular el valor absolut d'un número. S'indica com a |x|. El valor absol ut és la part sencera POSITIVA. Per exe mple:

|3| = 3

|-3| = 3

Dades d'entrada del problema:

- El número al que volem aplicar el v alor absolut

Sortida: el número en valor absolut

D) Algorisme del classificador d'edats

Problema: Saber el preu d'una entrada a un museu.

Dades d'entrada del problema:

- Edat del client

Detalls:

Per saber el preu s'aplica la següent regl a:

Menors de 3 anys: 2€

Menors de 10 anys: 10€

A partir de 10 anys: 25€

Sortida: el cost de l'entrada

b) Exercicis de diagrames de flux / quadres de decisió

D.bis) Algorisme del classificador d'edats
Problema: Saber el preu d'una entrada a un museu.
Dades d'entrada del problema:

- Edat del client
- Si té o no carnet d'estudiant

Detalls:
Per saber el preu s'aplica la següent regla:
Menors de 3 anys: 2€
Menors de 10 anys: 10€
A partir de 10 anys: 25€

En el cas de tenir carnet d'estudiant s'aplica un 10% de descompte **addicional**.

Sortida: el cost de l'entrada

b) Exercicis de diagrames de flux / quadres de decisió

C) Algorisme d'assignació de beques (fer diagrama de flux i quadre de decisió)

Problema: Calcular la quantitat de diners que podem donar a una persona en concepte de beca

Dades d'entrada del problema:

- Edat
- Ingressos familiar
- Número de germans

Detalls: Els sistema de beques té una sèrie de regles bàsiques.

Per a persones menors d'edat, les beques són més generoses. Per aquest motiu, en el cas de ser menor d'edat s'incrementa la beca aplicable en un 10%.

Per fer el càlcul de l'import becat, primer es verifica que els ingressos familiars no siguin superiors a la quantitat següent: (número de germans * 5700)+500

Si és superior o igual a aquesta, la beca queda anul·lada (0€), excepte en el cas de que siguin més de 4 germans, en el que es dona una quantitat compensatòria de 150€.

Quan la xifra d'ingressos familiars està per sota del límit, s'aplica la fórmula estàndard de beca:

$$\text{Beca} = 4000 * \frac{(\text{número de germans} * 5700) + 500 - \text{ingressos familiars}}{(\text{número de germans} * 5700) + 500}$$

Sortida: Quantitat de diners de la beca

b) Exercicis de diagrames de flux / quadres de decisió

D) Algorisme de la hipoteca (fer diagrama de flux i quadre de decisió)
Problema: Concessió d'hipoteca
Dades d'entrada del problema:

- Import préstec
- Ingressos anuals de la família
- Cost taxat del pis
- Termini de pagament (en anys)

Detalls: Per a concedir un préstec, el banc estudia cada cas, i té una sèrie de restriccio ns abans de donar la hipoteca. Per començar, s'exigeix que l'import demanat com a pré stec sigui com a màxim el 80% del cost taxat del pis. Si no és així, no es concedirà el pr éstec. Hi ha una segona condició que cal complir, l'import mensual del rebut no ha de su perar el 50% dels ingressos mensuals de la família per tal de garantir la subsistència.
Per calcular l'import a pagar mensual hi ha la següent fórmula (aproximada ☺):

Import mensual del rebut = Import del préstec*2 / (Termini de pagament *12)

Si s'acompleixen les dues condicions, es concedirà la hipoteca.

Sortida:

- Hipoteca concedida (si/no)
- Import mensual del rebut

b) Exercicis de diagrames de flux / quadres de decisió

E) Algorisme
Problema: Determinar quin és el major de tres números introduïts per teclat.
Dades d'entrada del problema:

- Ens subministren 3 números. (num1, num2, num3)

Sortida:

- Imprimirem per pantalla el número major dels tres introduïts. Per exemple, si l'usuari escriu 4,7 i 2, per pantalla treurem el 7.

c) Iteradors:

per (o "des de" o for)

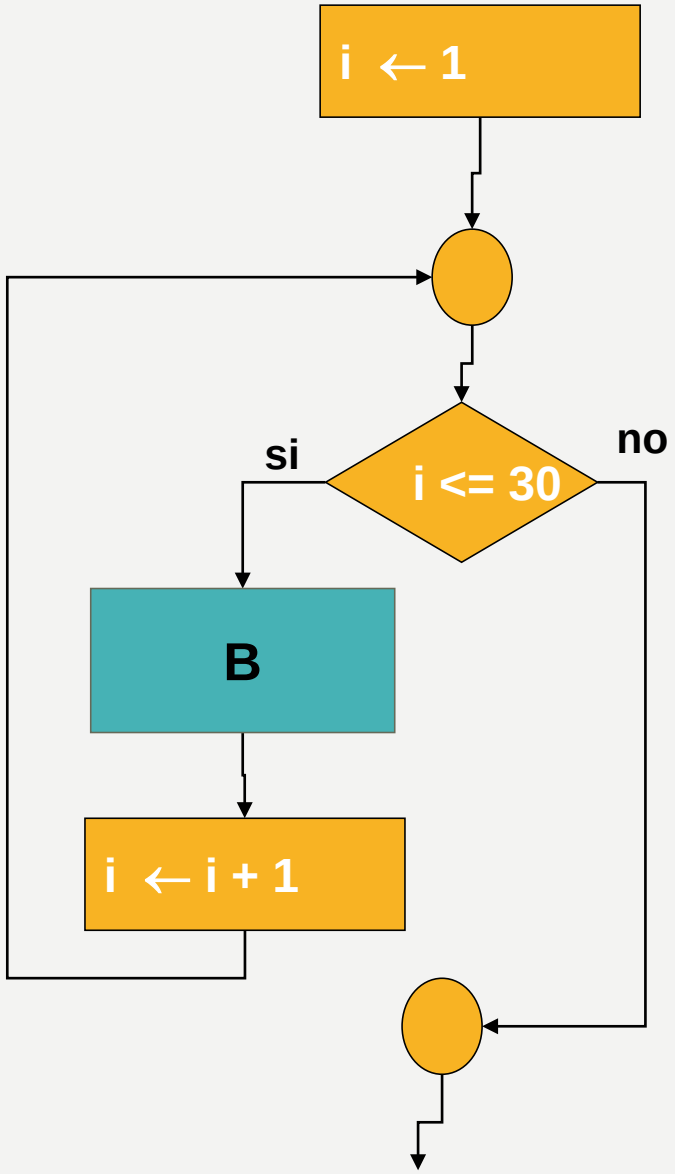
variable comptador *límit*

per i=1 fins a 30 fes

B

fper

≡



la variable comptador s'autoincrementa !

Bloc de línies de codi

(1 o més línies de codi , per exemple:

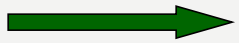
Llegir numero sencer → num
ACUMULAT ← ACUMULAT + num....

)

c) Iteradors : *per (o "des de" o for)*:

Exemple 1:

```
per i=1 fins a 3 fes
        escriu i
        escriu "Hola"
        escriu i+4
fper
```



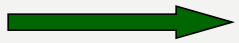
En pantalla

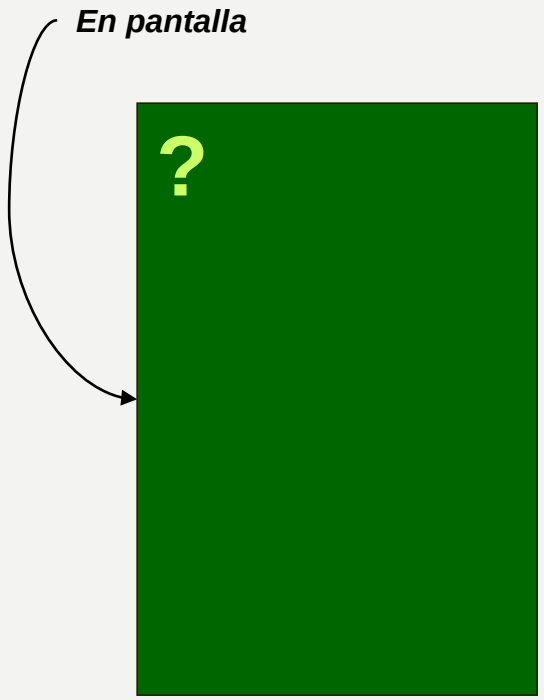
```
1
Hola
5
2
Hola
6
3
Hola
7
```

c) Iteradors : *per (o "des de" o for)*:

Exemple 2:

```
per i=1 fins a 3 fes
        escriu i
        i ← i +1
fper
```





UF1: Programació estructurada

1.- Elaborar un diagrama de flux que mostri els números parells compressos entre 10 i 20 ambdós inclosos.
2.- Fer un diagrama de fluxe que mostri la suma dels 20 primers números.
3.- Fer un diagramma de fluxe que llegeixi del teclat 10 números i imprimeixi només els números positius.

c) Iteradors: *mentre (while)*

UF1: Programació estructurada

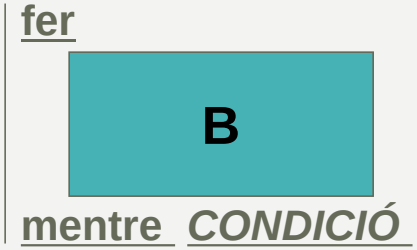
.1- Escriu un algortime que llegeixi 15 números negatius i el converteixi a convertirlos a positis i imprimeixi aquest números.

2.- Escriu un algoritme en diagrama de flux que imprimeixi els números compresos entre 1000 i 2000.

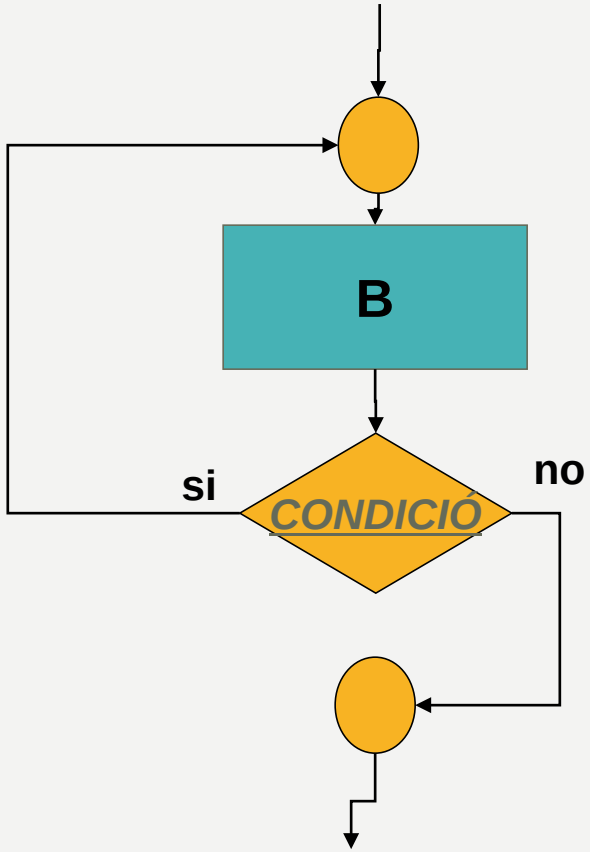
c) Iteradors

La tercera forma d'iteració és

Fer-mentre (do-while)



≡



ATENCIÓ: Sempre executarem com a mínim una vegada el Bloc de codi !!!!

1.- Escriu un algoritme en diagrama de flux que **imprima la suma de tots els números senars compresos entre el 1 i el 100.**