

Tema 1

Dades simples.

Instruccions seqüencials

- 1 Introducció
- 2 Dades simples
- 3 Expressions
- 4 Operadors
- 5 Funcions

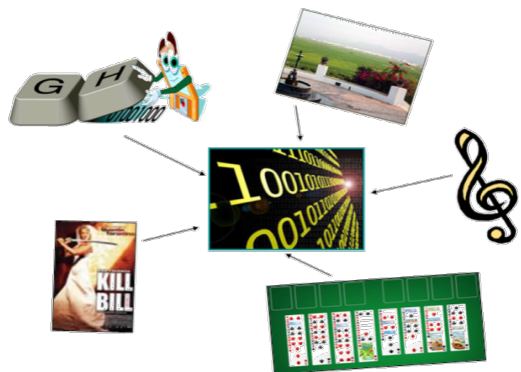
1 Introducció

Un programa és una seqüència d'instruccions que manipulen unes dades per a obtenir uns resultats.

DADES → PROGRAMA → RESULTATS

Eixes instruccions són ordres que li fem a l'ordinador. Per a això cal dir-li-ho en el llenguatge que entén, que és el llenguatge màquina, compost per seqüències de 0s i 1s, igual que tota la informació que es guarda en un ordinador (números, text, fotos, música, jocs, pel·lícules...):

Però com per a nosaltres (els humans) ens resulta molt difícil, li ho direm en altre llenguatge. Començarem amb Python i més avant vorem Java.



En este tema vorem les dades que són manipulades pels programes.

2 Dades simples

Una dada és qualsevol informació amb la qual treballa un algorisme.

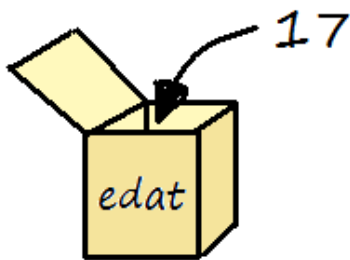
Cada dada és d'un tipus determinat que, bàsicament, serà enter, real, caràcter o lògic, però que dependrà del llenguatge de programació en què estem treballant.

Les dades apareixen en un programa en una de les següents formes:

- variables
- constants (simbòliques i literals)

2.1 Variables

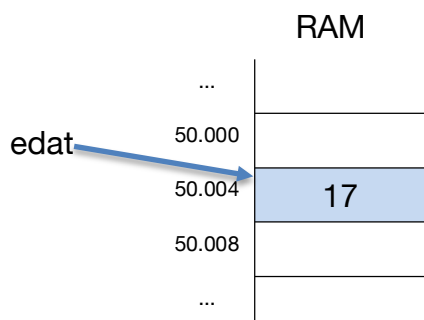
Una variable és un lloc on podem guardar una dada.



El dibuix representa la variable edat, que guarda el valor 17.
Cada variable es caracteritza per tindre:

- Un **nom** (edat) i un **tipus** (enter), que s'han d'especificar quan es defineix la variable en un programa, amb una instrucció declarativa (encara que hi ha llenguatges, com Python, que no cal indicar el tipus).
- Un **valor** (17), que li s'assignarà en alguna instrucció d'assignació (o bé en la mateixa instrucció declarativa) i que podrà ser canviat per altre valor les voltes que calga.

Les variables s'emmagatzemen en la memòria RAM, de forma que:



- El **nom** (edat) representa l'adreça de la RAM on està el valor
- El **tipus** (enter) especifica la quantitat de bytes necessaris per a guardar un valor (4).
- El **valor** és el contingut (17).

Exemple:

```
// instrucció declarativa
int edat;           // Definim una variable, de nom edat i de tipus enter

// instruccions d'assignació
edat = 17;          // Donem valor 17 a la variable edat
lleg(edat);         // Assignem per teclat un valor a edat. Per exemple, 19
edat = edat + 3;     // Tornem a canviar el valor. Ara valdrà 22
edat = 23.5;         // Error. Per què?

// utilització de la variable
escriu(edat)        // Escrivim en pantalla el valor de la variable edat
```

Nota: més endavant vorem estes instruccions detalladament.

2 Constants

Una constant és com una variable però que el valor no canvia durant l'execució del programa.

Les constants poden aparèixer en forma de literals o bé amb nom (constants simbòliques):

EXEMPLES DE CONSTANTS	
SIMBÒLIQUES	LITERALS
MAX_EDAT	99
PI	3'1416
VALOR_EURO	166'386
NOM_INSTITUT	"Jaume II el Just"
CICLES_INFORMATICA	true
MAJORIA_EDAT	18

Les constants simbòliques, igual que les variables, tenen un valor concret que se li dóna al principi del programa però, com ja hem dit, no poden canviar de valor.

Les constants literals alfanumèriques han d'expressar-se tancades entre cometes.

3 Expressions

Les constants i variables no apareixen aïllades, sinó formant part d'expressions. Una expressió és un càlcul necessari per a obtenir un resultat.

Una expressió és una combinació d'operands units mitjançant operadors.

- Els operands poden ser de diferents tipus:

- Literals: "Jaume II el Just", 100
- Constants: PI
- Variables: edat
- Funcions: arrel(100), longitud(nom)

- Els operadors els vorem en altre apartat

Exemples d'expressions:

- Numèriques:

edat

5

$2 * PI * \text{quadrat}(\text{radi})$

$(-b + \text{arrel}(\text{quadrat}(b) - (4 * a * c)) / (2 * a)$

- Alfanumèriques:

"Neus"

"Miquel " + "Garcia " + "Marqués"

- Lògiques:

true

false

valor1 < valor2

$(\text{valor1} < \text{valor2}) \&\& (\text{valor2} < \text{valor3})$

4 Operadors

Són els símbols de les operacions amb els quals es construeixen les expressions.

Depenent del tipus de dades dels operands o del tipus del resultat, tenim uns tipus d'operadors: aritmètics, lògics, relacionals i alfanumèrics

4.1 Operadors aritmètics

Són les operacions matemàtiques. Les variables o constants que hi intervenen són numèriques (enters o reals) i el resultat també. Els més usuals són:


OPERADOR	SIGNIFICAT
\wedge	Exponenciació
$*$	Producte
$/$	Divisió
$\%$	Residu de divisió entera
$+$	Suma
$-$	Resta

Regles de prioritat

Les expressions que tenen 2 o més operands necessiten unes regles que permeten determinar en quin ordre s'avaluen. Per exemple, si escrivim:

escriu($2*5-3$);

Què mostrarà? 7 o 4? La resposta és 7, ja que les regles de prioritat indiquen que l'operador del producte té més prioritats que el de la resta, com veiem a la taula.

OPERADOR	PRIORITAT
\wedge	Alta  Baixa
$*$ $/$ $\%$	
$+$ $-$	

Si 2 operadors d'igual prioritats coincideixen en una mateixa expressió, s'avaluaran d'esquerra a dreta. Però si volguérem canviar l'ordre d'avaluació en una expressió, utilitzarem els parèntesis necessaris. A banda, és recomanable l'ús de parèntesis davant del dubte.

4.2 Operadors relacionals

Servixen per a comparar 2 expressions, retornant un valor lògic: vertader o fals.

OPERADOR	SIGNIFICAT
<	Menor
>	Major
==	Igual
!=	Distint
<=	Menor o igual
>=	Major o igual

Per exemple, si $x=10$ i $y=20$, tenim que:

EXEMPLES D'EXPRESSIONS LÒGIQUES	VALOR
$(x+y) < 20$	false
$(y-x) <= x$	true
$(y-x) >= x$	true
$x == y$	false
$x != y$	true

També podem comparar caràcters (van entre cometes):

EXEMPLES D'EXPRESSIONS LÒGIQUES	VALOR
'c' < 'f'	true

4.3. Operadors lògics

Els operadors lògics són NO, I i O. Però per seguir la nomenclatura estàndard dels algorismes utilitzarem els noms anglesos: NOT, AND i OR. Actuen sobre operands o expressions lògiques i el resultat també és un valor lògic, que ve donat per les corresponents taules de veritat, on V és Vertader (true) i F és Fals (false):

x	NOT x
F	V
V	F

x	y	x AND y
F	F	F
F	V	F
V	F	F
V	V	V

x	y	x OR y
F	F	F
F	V	V
V	F	V
V	V	V

Per exemple:

NOT (3 < 5) → F

(3 < 5) AND (4 < 2) → F

(3 < 5) OR (4 < 2) → V

Expressions sinònimes:

NOT (a < b) → a >= b

NOT (a <= b) → a > b

NOT (a + b == 0) → a + b != 0 Compte! No canvien els operadors aritmètics.

NOT (true) → false

NOT (false) → true

NOT (jubilat == true) → NOT (jubilat) → jubilat == false

NOT (jubilat == false) → jubilat == true → jubilat

Lleis de De Morgan

Són regles que permeten transformar expressions lògiques en altres equivalents. Són molt útils per a simplificar expressions condicionals. Concretament, transformen expressions formades amb un NOT sobre alguna expressió que té dins algun AND, OR o NOT.

1a llei de De Morgan

$$\text{NOT}(A \text{ AND } B) \rightarrow \text{NOT}(A) \text{ OR } \text{NOT}(B)$$

Veiem-ho amb un exemple. Si A és "Plou", i B és "Fa fred":

$\text{NOT}(A \text{ AND } B)$ significa "No és cert que ploua i faça fred alhora"

Això, segons esta llei, és equivalent a dir:

$\text{NOT}(A) \text{ OR } \text{NOT}(B)$, que significa "No plou o no fa fred"

Veiem-ho d'una altra forma, amb les variables *plou* i *fred*:

$$\text{NOT}(\text{plou AND fred}) \rightarrow \text{NOT}(\text{plou}) \text{ OR } \text{NOT}(\text{fred})$$

És a dir, tinguen els valors que tinguen *plou* i *fred* (verdader o fals), sempre tindrem el mateix resultat en les 2 expressions equivalents. Comprovem-ho amb la taula de veritat:

p (plou)	f (fred)	p AND f	NOT(p AND f)	NOT(p)	NOT(f)	NOT(p) OR NOT(f)
V	V	V	F	F	F	F
V	F	F	V	F	V	V
F	V	F	V	V	F	V
F	F	F	V	V	V	V

2a llei de De Morgan

$$\text{NOT}(A \text{ OR } B) \rightarrow \text{NOT}(A) \text{ AND } \text{NOT}(B)$$

Veiem-ho amb un exemple. Si A és "Plou", i B és "Fa fred":

$\text{NOT}(A \text{ OR } B)$ significa "No és cert que: plou o faça fred"

Això, segons esta llei, és equivalent a dir:

$\text{NOT}(A) \text{ AND } \text{NOT}(B)$, que significa "No plou i no fa fred"

Veiem-ho ara amb les variables *plou* i *fred*:

$$\text{NOT}(\text{plou OR fred}) \rightarrow \text{NOT}(\text{plou}) \text{ AND } \text{NOT}(\text{fred})$$

És a dir, tinguen els valors que tinguen *plou* i *fred* (verdader o fals), sempre tindrem el mateix resultat en les 2 expressions equivalents. Comprovem-ho amb la taula de veritat:

p (plou)	f (fred)	p OR f	NOT(p OR f)	NOT(p)	NOT(f)	NOT(p) AND NOT(f)
V	V	V	F	F	F	F
V	F	V	F	F	V	F
F	V	V	F	V	F	F
F	F	F	V	V	V	V

3a llei. Doble negació

$$\text{NOT}(\text{NOT}(A)) \rightarrow A$$

Per exemple, dir que "No és cert que no plou" és el mateix que dir que "plou".

Com aplicar estes lleis quan el NOT actua sobre més d'un operador lògic?

Per exemple:

NOT (plou AND NOT(fred) AND sol AND humitat)

Caldria:

- Llevar el NOT que abarca tota l'expressió
- Canviar els AND per OR i al revés.
- Posar un NOT a cada part.

És a dir:

~~NOT~~ (NOT(plou) OR NOT(NOT(fred)) OR NOT(sol) OR NOT(humitat))

I, aplicant la llei de la doble negació, quedaria:

NOT(plou) OR fred OR NOT(sol) OR NOT(humitat)

Compte! Si en l'expressió que abarca el NOT hi ha ORs i ANDs (les 2 coses alhora), cal anar en compte en la transformació, ja que, com anem a veure a continuació, l'AND és més prioritari que l'OR. La solució seria, abans de fer la transformació, posar els parèntesis que calguen, i després ja fer la transformació conservant els parèntesis.

Per exemple, si tenim:

NOT (plou AND fred OR sol)

Primer posarem parèntesis per tindre una solució equivalent. Com l'AND és més prioritari que l'OR, posarem els parèntesis així:

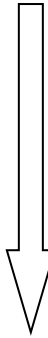
NOT ((plou AND fred) OR sol)

Ara ja apliquem les lleis de De Morgan, conservant els parèntesis de dins:

(NOT(plou) OR NOT(fred)) AND NOT(sol)

Regles de prioritat

Com els operadors lògics i relacionals poden formar expressions juntament amb els aritmètics, també necessitem unes regles de prioritat per a saber quins operadors s'avaluen primer.

OPERADOR	PRIORITAT
NOT	Alta  Baixa
^	
*, /, %	
+, -	
<, >, <=, >=	
==, !=	
AND	
OR	

Estes regles són bastant estàndards però podria dependre de cada llenguatge de programació.

No obstant, davant del dubte, sempre podem (i devem) utilitzar els parèntesis.

5 Funcions

Són trossos de codi que podem utilitzar en els nostres programes. Hi ha de 2 tipus: predefinides i definides per l'usuari.

5.1 Funcions predefinides

Els llenguatges de programació tenen funcions predefinides amb les quals podem dur a terme les tasques més usals. Les funcions (igual que en les de les matemàtiques) solen rebre un o més arguments i retornen un valor que anomenem resultat.

Per exemple, per a mostrar coses per pantalla tenim:

```
printf("Hola");           // en llenguatge C
System.out.println("Hola"); // en llenguatge de programació Java
escriu("Hola")            // forma que utilitzarem en els algorismes
```

5.2 Funcions definides per l'usuari

Podem crear funcions i usar-les en diferents parts del programa:

```
Programa principal {
    escriu("L'àrea del triangle de base 2 i altura 4 és:");
    area = areaTriangle(2, 4);
    escriu( area );
    escriu("L'àrea del triangle de base 3 i altura 6 és:" + areaTriangle(3, 6));
}
// -----
funció areaTriangle(base: enter, altura: enter) {
    area real;
    area = base * altura / 2
    retorna area;
}
```

Ja vorem en detall l'ús de funcions més endavant.

Exercicis

1. Calcula el valor de cada expressió si és vàlida. Si no és vàlida, indica el motiu.

- a) $10 * 3 + 5 * 2$
- b) $15 \% 4$
- c) $2 + 7 / 3$
- d) $4 + \text{"preu"}$
- e) $(5 + 2) < 8$
- f) $4 \geq 4$
- g) true OR false
- h) $5 \text{ OR } (2 < 3)$
- i) $(6 \geq 2) \text{ OR } (3 \leq 5)$
- j) $\text{NOT (NOT (NOT (4 < 10)))}$
- k) $4 + \text{false}$
- l) $4 + 2 * 4 / 2$
- m) $((5 < 0) \text{ AND } (6 \geq 7)) \text{ OR } (45 \% 5 \leq 0)$
- n) $((10 - 4) > 0) \text{ OR true}$
- o) $((10 - 4) < 0) \text{ OR true}$

2. Donats els següents valors de les variables $X=1$, $Y=4$, $Z=10$ i la constant $\text{PI}=3.14$, avalua les expressions següents:

- a) $2 * X + 0.5 * Y - 1 / 5 * Z$
- b) $((\text{PI} * X^2) > Y) \text{ OR } ((2 * \text{PI} * X) \leq Z)$
- c) $\text{"Hola, món!" == "Hola," + "món!"}$
- d) 'a' == 'A'

3. Construeix expressions correctes per a les fórmules següents:

- a) $ax^2 + bx + c \geq 0$
- b) $\frac{3x - y}{z} - \frac{2xy^2}{z - 1} + xy$
- c) $\frac{a}{b - \frac{c}{d - \frac{e}{f - g}}} + \frac{h + i}{j + k}$

4. A partir de les següents constants:

gran = fals; redó = cert; suau = fals;

...indica quin serà el valor de les següents expressions:

- a) gran i redó i suau;
- b) gran o redó o suau;
- c) gran i redó o suau;
- d) gran o redó i suau;
- e) gran i (redó o suau);
- f) (gran o redó) i suau;

5. Indica amb parèntesis l'ordre en què l'ordinador executaria les diferents operacions.

- a) $x + y + z$
- b) $x * y + z$
- c) $x + y * z$
- d) $x - y * z$
- e) $x + y / z$
- f) $x * y / z$

g) $x / y / z$

h) $x / y * y + x \% y$

i) $x / y + z + x$

6. Transforma les següents expressions en altres equivalents utilitzant les lleis de De Morgan. Cal tindre en compte que a, b, c són variables enteres i p, q, r són variables booleanes (lògiques).

a) $\text{NOT} ((p \text{ AND } q) \text{ OR } r)$

b) $\text{NOT} ((a == b) \text{ OR } (a == 0))$

c) $\text{NOT} (\text{NOT } p \text{ OR } \text{NOT } q \text{ OR } (a == b + c))$

d) $\text{NOT} (p \text{ AND } (q \text{ OR } r))$

e) $\text{NOT} ((a < b) \text{ AND } (b < c))$

f) $\text{NOT} (\text{NOT } p \text{ AND } q \text{ OR } \text{NOT } r)$

g) $\text{NOT} (\text{NOT} (a <> b) \text{ OR } (a + b == 7))$

h) $\text{NOT} ((a / b == 0) \text{ OR } (a == c))$

7. Donats els valors inicials de les següents variables enteres i lògiques:

a=3; b=5; c=7; p=cert; q=fals;

...indica els valors que tindran estes variables després de les següents assignacions.

Nota: en cada apartat es tindrà en compte els canvis de les variables dels apartats anteriors.

a) $a = 3 * b;$

b) $b = a + c;$

c) $p = p \text{ and } (c > b);$

d) $q = p \text{ or } q;$

- e) $r = a == b$;
- f) $a = a + 1$;
- g) $b = b - 2$;
- h) $a = a$;
- i) $b = b / 2 + c \% 3$;

8. Sent a, b, c, d variables numèriques, escriu l'expressió lògica corresponent a:

- a) Els valors de b i c són tots dos superiors al valor de d
- b) a, b i c són idèntics
- c) a, b i c són idèntics però diferents de d
- d) b està comprés, estrictament, entre els valors de a i c
- e) b està comprés, estrictament, entre els valors de a i c , i el valor de a és més xicotet que el valor de c
- f) Hi ha, com a mínim, dos valors idèntics entre a, b i c
- g) Hi ha dos valors idèntics entre a, b i c , i només dos.
- h) Hi ha, com a màxim, dos valors idèntics entre a, b i c

9. Escriu l'expressió algorísmica de les següents expressions:

- a) Avaluar si el contingut d'una variable numèrica és divisible per 10 o per 7.
- b) Avaluar si una variable *preu* no és menor de 100 € ni major de 200 €.

10. Si DN, MN, AN representen el dia, mes i any d'una persona i DA, MA, AA el dia, mes i any actuals, expressa amb una expressió si la persona ha complit 18 anys.

11. En un algorisme que analitza els resultats d'exàmens, hi ha 5 variables definides:

```
char opcio;      // Tipus d'alumne: (C)iències o (L)etres
```

```
int nl, nv, nm, nf; // Notes de literatura, valencià, mate i física d'un alumne
```

Totes les notes estan calculades sobre 10 i tenen el mateix pes per a fer la mitjana.
Escriu les expressions lògiques corresponents a:

- a) La mitjana de les quatre notes és superior a 5
- b) Les notes de mate i valencià són superiors a la mitjana de les quatre notes
- c) Hi ha, com a mínim, una nota superior a 5
- d) Totes les notes són superiors a 5
- e) La mitjana de les quatre notes és superior o igual a 5, i la mitjana de les notes de l'opció que ha agafat l'alumne també.