

# FONAMENTS D'ORDINADORS

## TEMA2: Programació Bàsica

Manel Guerrero

## [H1] Variables

- Objecte identificat amb un nom, que és d'un tipus determinat (ex: enter) i que guarda un valor que es pot anar variant i consultant.
- `tipus_var nom_var = valor_inicial;`
  - `int i; /* declaració */`
  - `int i, j, k; int a=2, b=3;`
  - `i = 0; /* assignació d'un valor */`
  - `int i = 0; /* inicialització (declaració + assignació d'un valor inicial) */`

## Tipus de variables

- Caracter:
  - `char c = 'A';`
- Entera:
  - `int i = 7;`
- Coma flotant:
  - `float f = 7.5;`
- Coma flotant d'alta precisió:
  - `double pi = 3.141592653589793;`

## Nom d'una variable

- Compost per lletres, números i “\_”.
- No pot començar per número.
- “Case sensitive” (diferència entre majúscules i minúscules). `Var`, `VAR`, `var` són noms de variables diferents.
- Típicament tot en minúscules i usant el “\_” en lloc de l'espai. Ex: “`nom_variable`”.

## Constants

- Identificador que representa un valor constant. Es pot consultar però no modificar.
- `#define nom_constant valor_constant`
  - `#define TRUE 1 /* Enter */`
  - `#define FALSE 0`
  - `#define PI 3.1415 /* Decimal */`
  - `#define YES 'Y' /* Caràcter */`
- Les línies de codi que comencen amb '#' no acaben en ';'.
- Els noms de les constants segueixen les mateixes normes que els de les variables.
- Però, típicament tot en majúscules i usant el "\_" enlloc de l'espai.

## Referencia

- Les transparències que venen a continuació (fins arribar a [H2]) són més de referencia per més endavant que una altra cosa.
- Serà quan veiem una cosa anomenada «sentències condicionals» que ens seran útils els operadors relacionals i els operadors lògics, juntament amb la prioritat d'operadors.
- Veiem no obstant els exemples de T2H1.

## Expressions

- Són formules formades per variables, constants, operadors i parèntesis que es poden avaluar donant com a resultat un valor.
- Operadors:
  - Aritmètics:
    - Suma [ + ] i Resta [ - ]
    - Multiplicació [ \* ] i Divisió [ / ]
    - Residu [ % ]
    - Increment [ ++ ]
    - Decrement [ -- ]
  - Relacionals:
    - Més gran que [ > ]
    - Més petit que [ < ]
    - Igual que [ == ]
    - Diferent que [ != ]
    - Més gran o igual [ >= ]
    - Més petit o igual [ <= ]
  - Lògics:
    - AND [ && ]
    - OR [ || ]
    - NOT [ ! ]

## Expressions: Exemples

- Fals = 0. Cert != 0
- Longitud cercle:  
`#define PI 3.1415`  
`float radi=3, lc;`  
`lc = PI * 2 * radi;`
- El punt (x,y) pertany a cercle de radi 'r'?  
`int p;`  
`float x=1, y=-2, r=2.23;`  
`p = (x*x + y*y == r*r);`
- Àrea triangle:  
`float b=2, a=3, at;`  
`at = b*a/2;`- És 'n' parell?  
`int n = 3, p;`  
`p = (n%2 == 0);`- Les bruixes es pentinen?  
`#define TRUE 1`  
`int plou=TRUE, sol=TRUE;`  
`int lbep;`  
`lbep = plou && sol;`

## Prioritat d'operadors

Operator Type	Operator	Associativity
Primary Expression Operators	() [] . -> expr++ expr--	left-to-right
Unary Operators	* & + - ! ++expr --expr (typecast)	right-to-left
Binary Operators	* / % + - < > <= >= == != && 	left-to-right
Assignment Operators	= += -= *= /= %=	right-to-left

From: [www.swansontec.com/sopc.html](http://www.swansontec.com/sopc.html)

- Operators higher in the chart have a higher precedence.
- The "Associativity" column on the right gives their evaluation order.

## Exemples d'expressions

- `a * x * x + b * x + c /* paràbola */`
- `(b1 + b2) * h / 2 /* àrea trapezoide */`
- `!error && calcul_finalitzat`
- `int a=2, b=3;`
- `a++ + b++` retorna 5 (Trampa mortal!)
- `a <= b` retorna 1 (cert).
- `a == b` retorna 0 (fals).
- `(1 && 0) || (1 || 0)` retorna 1 (cert)

## Exemples d'expressions 2

- `int a=2, b=3; /* Enters */`
- `b / a` retorna 1 (Trampa mortal!)
- `float a=2, b=3; /* Coma flotant */`
- `b / a` retorna 1.500000
- `float a=2; int b=3; /* var!=0 és cert */`
- `b && a` retorna 1 (Perill!)

## Sentencies

- Una sentència ordena una acció.
- Les sentències d'assignació assignen un valor a una variable.
  - `nom_var = expressió;`
  - `i = 0;`
  - `y = a * x * x + b * x + c;`
- Totes les sentències acaben en “;”

## printf

- Per imprimir missatges per pantalla

```
#include <stdio.h>
```

```
int res;
```

```
res = 2 + 3;
```

```
printf("El resultat és %d\n", res);
```

- "\n" salt de línia | "\t" tabulat | "%%" %

```
printf("El %d %% de %d és %d\n",  
      percent, num, res);
```

- Començar el programa àrea d'un triangle.

## Exemple %%

```
#include <stdio.h>
```

```
#define PORCENTAJE 20 // Podriamos definirlo como 20.0 (float)
```

```
/* calcula el 20% de un valor */
```

```
main() {
```

```
    float dato, res;
```

```
    printf("Introduce un valor: ");
```

```
    scanf("%f", &dato);
```

```
    res = dato * PORCENTAJE/100;
```

```
    printf("El %f %% de %f es %f \n", PORCENTAJE, dato, res);
```

```
}
```

## printf: nivell pro

- %+d, %+f // mostra el signe sempre
- int a=64; printf("a=%6d \n",a);  
// escriu "a= 64"
- int hh=0, mm=3, ss=6; printf("%02d:%02d:  
%02d\n", hh, mm, ss);  
// escriu "00:03:06"
- %.2f : imprimeix el número amb dos decimals
- %8.2f : longitud total de 8 dígit, amb dos decimals

## scanf

- Per llegir dades per teclat:

```
#include <stdio.h>
```

```
int i;
```

```
float f;
```

```
scanf("%d", &i);
```

```
scanf("%f", &f);
```

- Completar el programa àrea d'un triangle (t2\_0\_area\_triangle.c).
- Felicitats! Ja hem fet el nostre primer programa. :-)

## scanf: nivell pro

- Si introduïm caràcters no numèrics retorna un -1, no s'assigna res a dada i els caràcters introduïts es llegiran de nou en el proper scanf. Trampa mortal!

```
#include <stdio.h>
```

```
int err, dada;
```

```
err = scanf("%d", &dada);
```

- Llegeix tres enters separats per espai i els assigna a i1, i2 i i3:

```
int i1, i2, i3;
```

```
scanf("%d %d %d", &i1, &i2, &i3);
```

## [H2] Tipus de dades

Declaració	Rang	printf/scanf
<b>unsigned char</b>	<b>[0, 255] (1 Byte)</b>	<b>%hhu</b>
unsigned short	[0, 65.535] (2 Bytes)	%hu
<b>unsigned int</b>	<b>[0, 2<sup>32</sup> -1] (4 Bytes)</b>	<b>%u</b>
unsigned long	[0, 2 <sup>64</sup> -1] (8 Bytes)	%lu
short	[-32.768, 32.767] (2 Bytes)	%hd
<b>int</b>	<b>[-2<sup>31</sup>, 2<sup>31</sup> -1] (4 Bytes)</b>	<b>%d</b>
long	[-2 <sup>63</sup> , 2 <sup>63</sup> -1] (8 Bytes)	%ld
<b>char</b>	<b>Taula ASCII (1Byte)</b>	<b>%c</b>
<b>float</b>	<b>[1.7E-38, 3.4E38] ( 4 Bytes)</b>	<b>%f %e</b>
<b>double</b>	<b>[2.22E-308, 1.79E308] (8 Bytes)</b>	<b>%lf %le</b>
long double	(16 Bytes)	%Lf %Le

## scanf: de diferents tipus

```
#include <stdio.h>
```

```
int i; char c; float f; double d;
```

```
unsigned int ui; unsigned char uc;
```

```
scanf("%d", &i); scanf("\n%c", &c);
```

```
scanf("%f", &f); scanf("%lf", &d);
```

```
scanf("%u", &ui); scanf("%hhu", &uc);
```

```
scanf("%*c"); // llegeix char i "el llença"
```

- El "\n" en el cas del char és per que si hi han espais en blanc o retorns de carro abans del caràcter se'ls salti (en el cas de "%d" i "%f" això es fa automàticament).

## Conversions de tipus

- Implícita
  - C permet assignar expressions d'un tipus a variables d'un altre tipus.
  - char <- int: c = 48; /\* c = 48 % 256\*/
  - int <- char: i = 'a'; /\* i = 97 \*/
  - int <- float: i = 3.5; /\* i = 3 (trunca) \*/
  - float <- double (arrodoneix)
- Explícita
  - int a=2, b=3; float f = (float)b / (float) a; // f és igual a 1.500000
  - float f = b / a; // f és igual a 1.000000

## Estructura d'un programa en C

- /\* llibreries \*/ (#include)
- /\* definició de constants \*/ (#define)
- /\* definició de tipus de dates \*/ (globals)
- /\* prototipus de funcions \*/ (???)
- /\* main : funció principal \*/ (principi)
- /\* definició funcions\*/ (???)

## Exercicis

- A classe. Fer un programa que:
  - Passi un caràcter de majúscules a minúscules.
  - Intercanviï el valor de 2 variables.
  - Roti el valor de 3 variables.
  - Converteix un caràcter numèric (char '0'..'9') a l'enter corresponent.
- A casa: Feu tots els problemes del printf i scanf. [Tema 2. Variables, expresiones y sentencia de asignación].

## De majúscules a minúscules

```
#include <stdio.h>

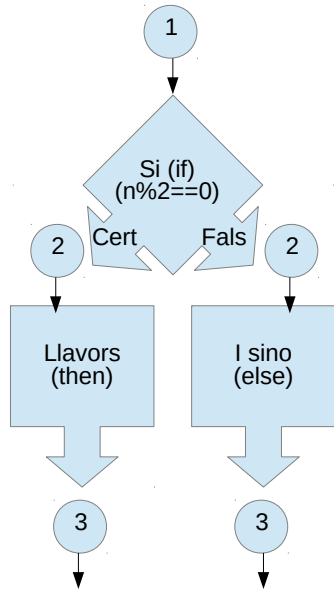
/* Pasar de minúsculas a mayúsculas: c = c -('a'-'A'); */
main() {
    char c;

    printf("Introduce un carácter: ");
    scanf("%c", &c);
    c = c + 'a' - 'A';
    /* c = (char) ( (int)c - (int)'a' + (int)'A' ) */
    printf("El carácter en minúscula es %c\n",c);
}
```

## [H3] Interludi

- Quanta gent no ha aconseguit compilar i executar els exemples?
- Quins errors de compilació us heu trobat?
- Quins programes heu provat de fer?
- Quins dubtes teniu de tot el que s'ha explicat fins ara?
- Quanta gent s'ha mirat la col·lecció de problemes?

## El desviador condicional



```
Si (passa això) llavors {  
    [ FAIG_AIXO ]  
} i sino {  
    [ FAIG_AIXO_ALTRE ]  
}
```

```
if (n%2 == 0) {  
    [ THEN ]  
} else {  
    [ ELSE ]  
}
```

## Sentencies condicionals: if

```
if (condition)  
    then_block;
```

```
if (condition)  
    then_block;  
else  
    else_block;
```

```
if (condition1)  
    then_statements1;  
else if (condition2)  
    then_statements2;  
else  
    else_statements;
```

- Si 'condition' és certa, llavors executo el bloc 'then'. Sinó és certa executo el bloc 'else'.
- Un 'else' correspon a l'últim 'if'.
- Usar claus quan un bloc (then o else) té més d'una instrucció.

## Sentencies condicionals: claus

```
if (condition) {  
    then_block;  
}
```

```
if (condition) {  
    then_block;  
} else {  
    else_block;  
}
```

```
if (condition1) {  
    then_statements1;  
} else if (condition2)  
{
```

```
    then_statements2;  
} else {  
    else_statements;  
}
```

- De fet, usar claus sempre evita problemes.

## Sentencies condicionals: exemples

```
if(c >= 'A' && c <= 'Z')  
    printf("Caràcter en  
majúscules\n");  
  
if(a%2 == 0)  
    printf("par ");  
else  
    printf("impar ");
```

```
if (c == 'y') {  
    printf("Yes\n");  
} else if (c == 'n') {  
    printf("No\n");  
} else {  
    printf("Error\n");  
    printf("y/n\n");  
}
```

## if: problemes

- Fer un programa que digui si el caràcter introduït és un dígit, un espai, una lletra majúscula, una lletra minúscula o cap de les anteriors.
- Demanar tres números “scanf(“%d %d %d”, ...” com les mides dels cantons d'un triangle i dir si el triangle és equilàter, isòsceles o escalè.
- Problemes de la col·lecció:
  - #2: Mostrar el valor màxim de dos enters.
  - #3: Mostrar el valor màxim de tres enters.
  - #5: Dir si un enter es parell o senar i el seu quadrat és menor estricta que 145000 o no.

## [H4] Problema: MCD

- Algorisme d'Euclides (MCD(a,b)):
  - Si/mentre b no val 0:  $MCD(a,b) = MCD(b,a\%b)$
  - Si/quan b val 0:  $MCD(a,0) = a$
  - Com ho podem fer?

MCD(654, 2322) A = 2322    b = 654 A = 654    b = 360 a = 360    b = 294 a = 294    b = 66 a = 66    b = 30 a = 30    b = 6 a = 6    b = 0 MCD = 6	MCD(a,b): a: 25 b: 10 a = 10    b = 5 a = 5    b = 0 MCD = 5	MCD(a,b): a: 10 b: 25 a = 25    b = 10 a = 10    b = 5 a = 5    b = 0 MCD = 5
--	---	---

## Sentències iteratives: while

```
while (expression)
    statement;

while (expression) {
    block;
}

int i;
i = 0;
while (i <= 10) {
    printf(“%d\n”, i);
    i++;
}
```

- Mentre l'avaluació de “expression” retorni cert (mentre “expression” sigui diferent de '0') executar “statement” i tornar a avaluar “expression”.
  - Equivalent amb for (per la pràctica de sentències iteratives):
- ```
int i;
for(i=0;i<=10;i++) {
    printf(“%d\n”, i);
}
```

## while: problemes

- Llegir en un bucle infinit enters i imprimir-los per pantalla.
  - Modificar el programa perquè si l'usuari introdueix caràcters no numèrics (trampa mortal del scanf) el programa es recuperi.

## [H5] while: problemes (2)

- Col·lecció de problemes:
  - #2: “Escriu un programa que mostri per pantalla la suma dels 20 primers números múltiples de 7 o 5.”
  - #3: Programa que demani un enter 'n' i retorni els primers 'n' elements de la serie de Fibonacci.
    - $[F(1) = 1, F(2) = 1, F(i) = F(i-1) + F(i-2)]$
    - si 'n' = 7 que retorni “1 1 2 3 5 8 13”.

## [H6] Problema amb el while

```
int n = -1;

while (n<0 || n>10) {
    printf("Num entre 0 y 10: ");
    scanf("%d", &n);
};

char c = '0';

while (c<'a' || c>'z') {
    printf("Car minuscules: ");
    scanf("%d", &n);
};
```

- Què passa quan la condició de permanència en el bucle depèn de una variable que s'inicialitza dins del bucle?
- Podem fer la 'xapussilla' de inicialitzar la variable a un valor que fa que la condició de permanència sigui certa la primera vegada que s'avalua.
- Però és una 'xapussilla'. I si la variable és una posició del vector el tema es complica. (Veurem vectors en el proper tema).

## Sentències iteratives: do while

```
do{
    statement;
} while (expr);

EQUIVAL A:
statement;
while (expr) {
    statement;
}

===
do {
    printf("Intro num entre 0 y 10: ");
    scanf("%d", &n);
} while (n<0 || n>10);
```

- Abans d'avaluar l'expressió 'expr' iteració, s'executa el cos del bucle una vegada.
- A partir de llavors funciona igual que un while.
- En el fons es com un while on tenim garantit que el cos del bucle s'executa com a mínim una vegada.
- Típicament s'usa per llegir valors de teclat i verificar que compleixen els requisits que volem que compleixin: que estiguin dins d'un rang determinat o que compleixin determinades condicions.

## Do while: problemes

- Fer un programa que vagi demanant una seqüència d'enters acabada en zero. Després de llegir cadascun dels enters digui quin és el total acumulat i en llegir el zero retorni la suma de tots els enters. [t2\_3\_do\_suma.c]
- Fer un programa amb un menú 's' suma, 'm' multiplica i 'a' acaba. [t2\_3\_do\_menu.c]

## [H7] Sentencias iteratives: for

```
for ([expr1]; [expr2]; [expr3]) {
    statement;
}
EQUIVAL A:
expr1;
while (expr2) {
    statement;
    expr3;
}
===
for (i=0; i<100; i++) {
    sum += x[i];
}
===
for (i=0, sum=0, sumsq=0; i<100; i++) {
    sum += i; sumsq += i*i;
}
===
for (i=0, sum=0, sumsq=0; i<100;
    i++, sum += i, sumsq += i*i);
```

- Abans de la primera iteració, expr1 és executada.
- “statement” és executat repetidament fins que expr2 val 0.
- Després de cada iteració expr3 és executada. Típicament s'usa per incrementar la variable comptador del bucle.
- Les 3 “expr” són opcionals.
  - for(;;) { statement; }
  - és un bucle infinit.
- Si expr2 és buida equival a '1'.

## for: trampes

- Incorrecte a C90 (correcte a C99):  
for (int i=0;i<=10;i++) {  
 ...  
}
- Al compilar ens dirà: "error: 'for' loop initial declarations are only allowed in C99 mode."
- <http://codingstack.com/questions/error-for-loop-initial-declarations-are-only-allowed-in-c99-mode>

- El que si que és correcte és declarar variables al obrir '{'. Tot i que no és recomanable! Ja sigui el '{' d'un 'for', 'while' o 'if':  
 int i;  
 for(i=0; i<=10; i++) {  
 int j;  
 for(j=0; j<=10; j++) {  
 printf("%d %d\n",  
 i, j);  
 }  
 }

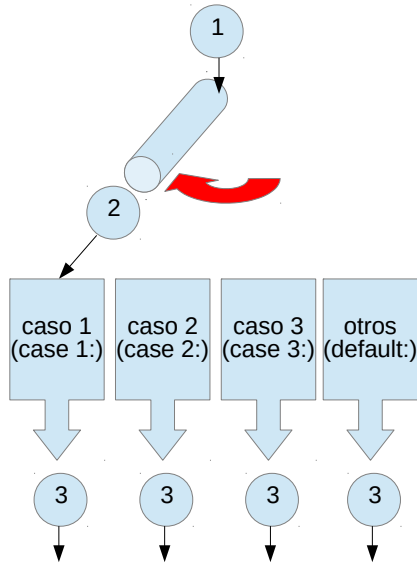
## for: problemes

- Treure per pantalla la taula de multiplicar del 7.
- Treure per pantalla les taules de multiplicar del 0 al 10.
- Treure per pantalla el mapa del Campus Nord.
- ...

## [H8] for: problemes (2)

- Col·lecció de problemes:
  - #5: “Escriba un programa que muestre por pantalla el valor máximo de una secuencia de 10 valores enteros introducidos por el usuario a través del teclado.”
  - #6: “Escriba un programa que lea desde teclado un número entero n. El programa luego deberá leer n números enteros y mostrar por pantalla el mayor y el menor valor de ellos.”
  - #15: “Considere la función  $f(x,y) = x^2 - 9xy + y^2$ , donde x, y son valores enteros entre -10 y 10. Escriba un programa que encuentre y muestre el valor máximo de f en ese intervalo.”

## [H9] El conmutador condicional



```
switch (n) {
  case 1:
    printf("uno\n");
    break;
  case 2:
    printf("dos\n");
    break;
  case 3:
    printf("tres\n");
    break;
  default:
    printf("no se\n");
}
```

## Switch ... case

```
switch ([expr]) {
  case constant1:
    Codi_n==constant2;
    break;
  case constant2:
    Codi_n==constant2;
    break;
  case [label]:
    ...
  default:
    Codi_n!=tots_cases;
}
```

- Label must be unique
- Labels must be a constant expression
- Label must be int or char
- Label cannot be floating point
- At most one default label
- Default label is optional
- Default can be placed anywhere
- Break leaves switch
- 2 or + cases may share one break
- Relational operators not allowed

## [H10] switch case problemes

- t2\_5\_switch\_menu.c
- t2\_5\_switch\_calc.c
- t2\_5\_switch\_dia.c