

Variables



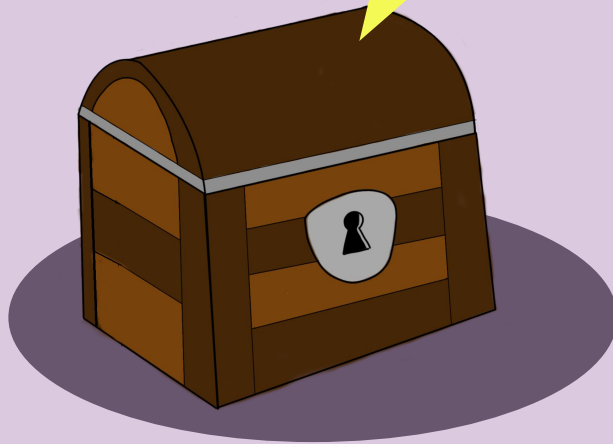
Programació

Xavier Sala Pujolar
Institut Cendrassos

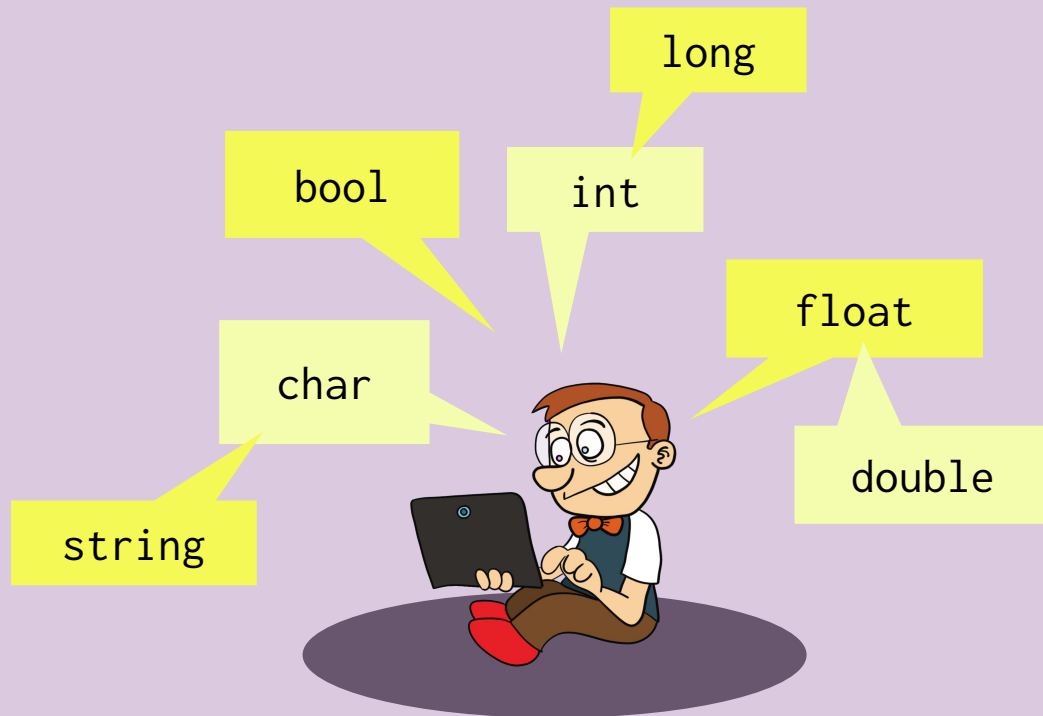


Una variable és **un lloc on es
guarden dades**

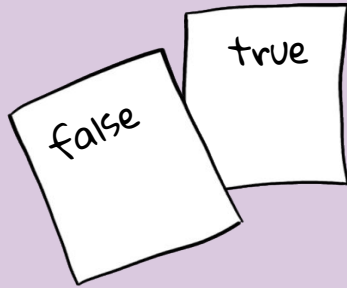
En C#, només dades
d'un determinat tipus



Els **tipus de dades bàsics** són
semblants als dels altres
llenguatges

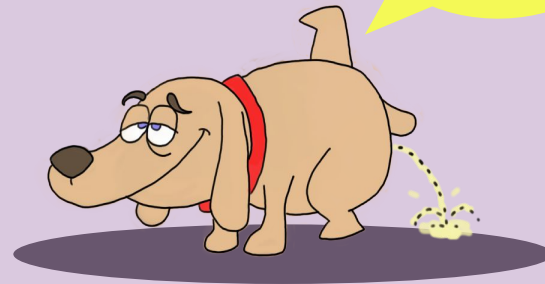


El tipus booleà només pot
tenir dos valors **true** o **false**
(que es poden interpretar,
si/no, 0/1, ...)

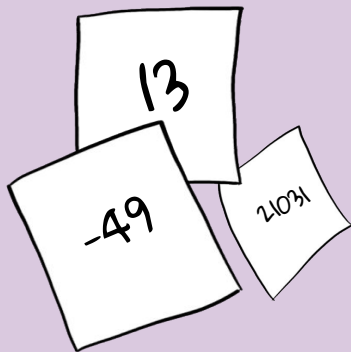


Vine cap aquí !

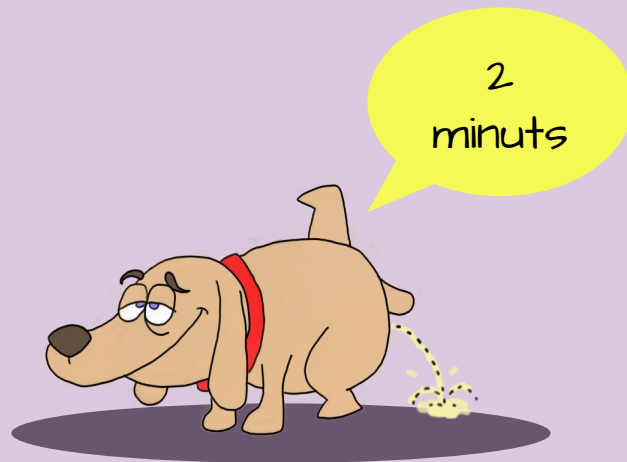
No



Els tipus **int** i **long** permeten
valors numèrics sense
decimals



Vine cap aquí!



Els números tenen una
capacitat màxima

uint

màxim número en 32 bits sense signe

11111111111111111111111111111111

0 ... 4.294.967.295

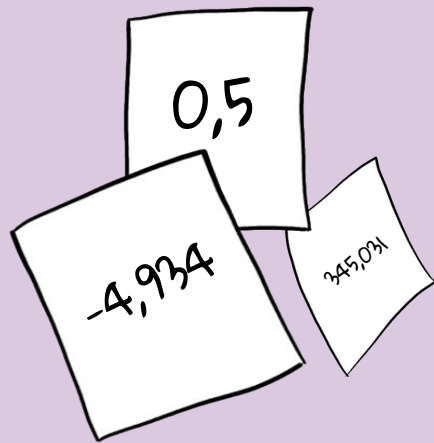
int

màxim número en 32 bits amb signe

01111111111111111111111111111111

-2.147.483.647 ... 2.147.483.647

Els números amb decimals
poden ser **float** o **double**

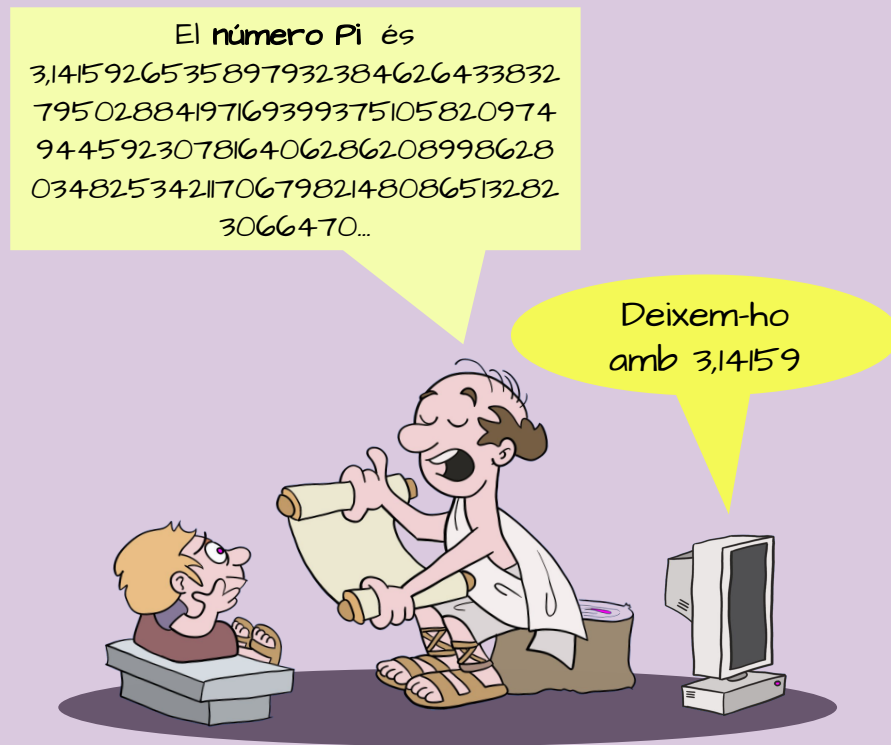


Vine cap aquí!

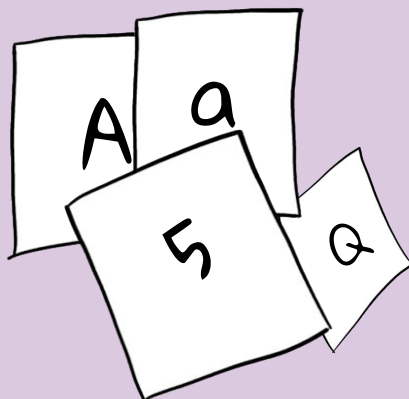


Els ordinadors **no poden**
representar tots els nombres
reals.

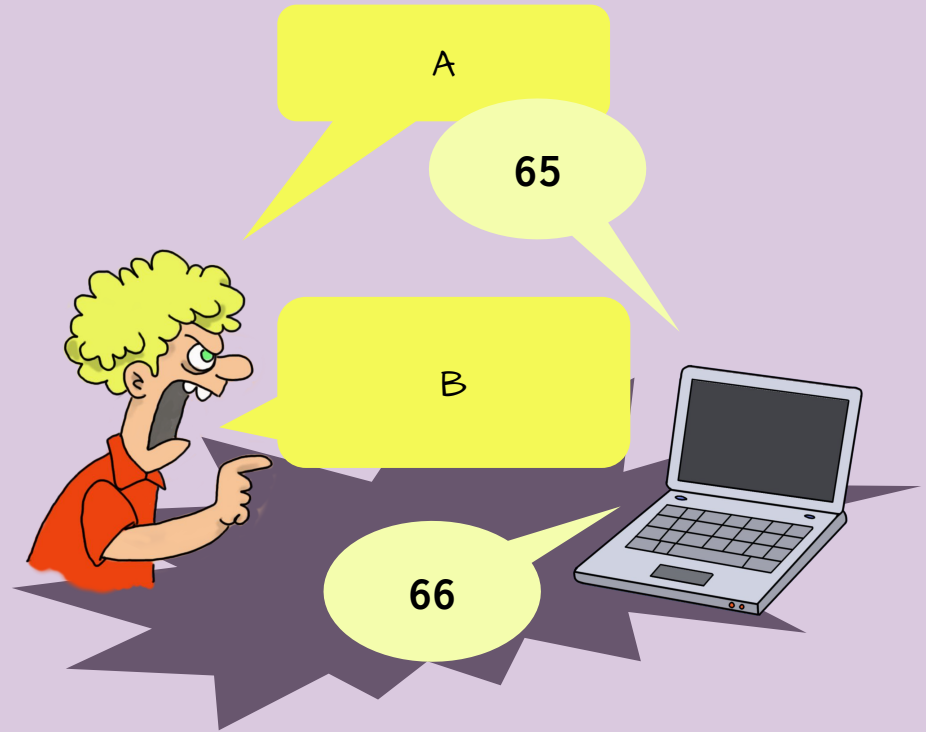
Fan servir una representació
aproximada



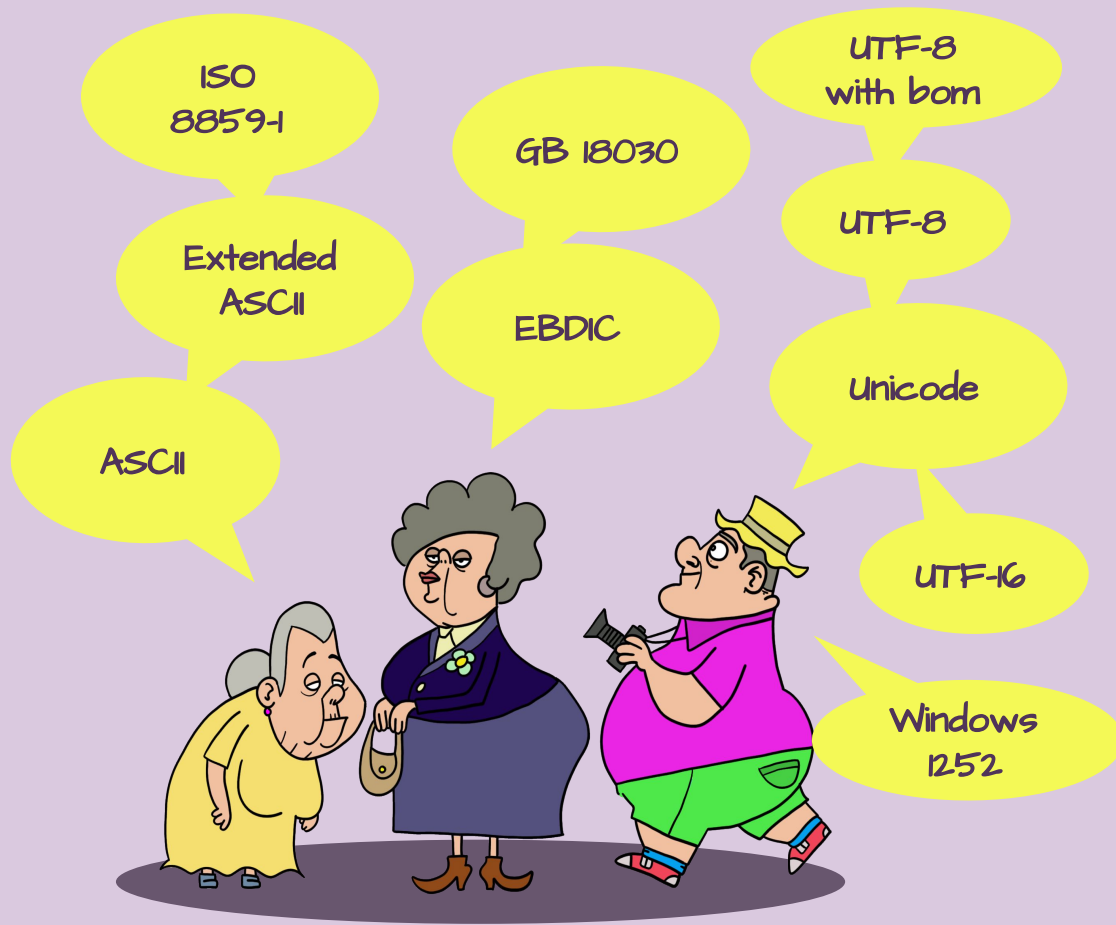
El tipus **char** només admet un sol caràcter



Els caràcters són números que
es codifiquen per representar
els caràcters



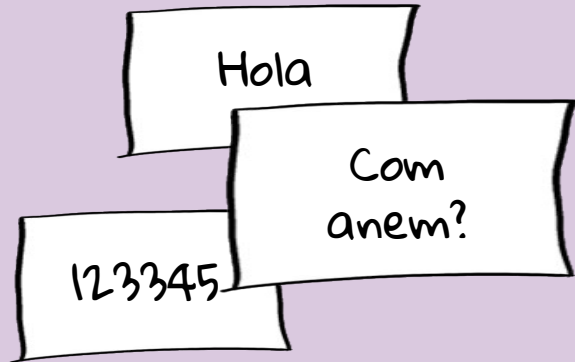
Hi ha molts tipus de
codificació de caràcters



L'estandard a la web i als
sistemes Unix sol ser **UTF-8**



El tipus **string** admet frases
formades per caràcters

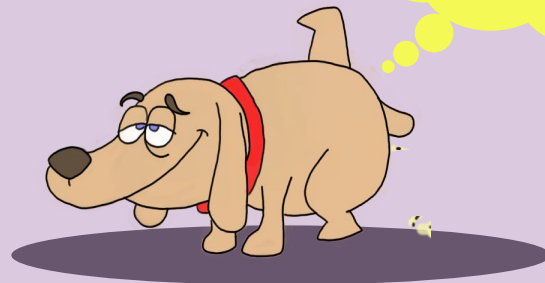


Hi ha un tipus de dades per
indicar que en la variable no
hi hauran dades: **el tipus
void**

No es fa servir en
variables. S'usa per indicar
que no es tornaran dades.

Vine cap aquí !

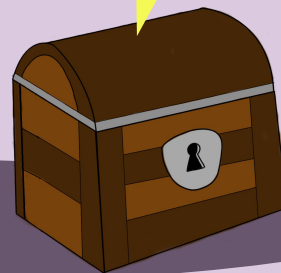
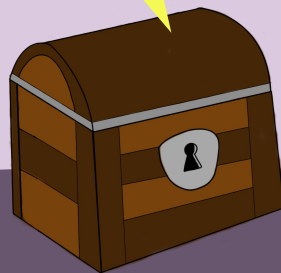
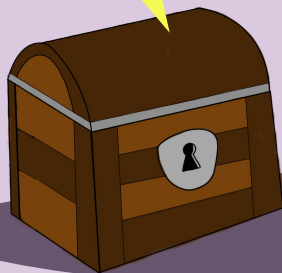
M'han passat
les ganes de
pixar



Caixa per
números

Caixa per
strings

Caixa per
números



Es defineix una
variable posant el
tipus i el **nom**

```
bool veritat;
```

El valor dels bool
és false

```
char lletraA;
```

```
int dos;
```

```
float tresimig;
```

Els números i char
són 0

```
string nom;
```

El valor dels
string és null

L'operador “**=**” permet assignar valor a les variables

```
bool veritat;  
veritat = true;
```

```
int numero;  
numero = 2;  
numero = 3 + numero;
```

```
string nom;  
nom = “Pere”;
```



Si ja en té el canvia

Es pot combinar la definició i l'operador per donar un valor inicial a les variables

```
bool veritat = true;
```

```
char lletraA = 'a';
```

```
int dos = 2;
```

```
int doble = 2;
```

```
int dobleDeDos = dos * doble;
```

```
float tresimig = 3.5;
```

```
string nom = "Pere";
```

Es poden definir
variables **deixant que**
el compilador decideixi
el tipus

```
var nom = "Pere";
```

Serà un string

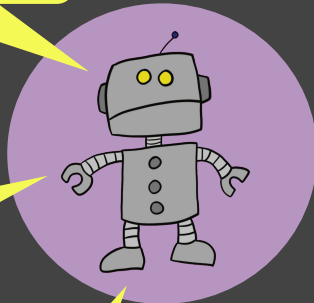
```
var numero = 2;
```

Serà un int

```
var numero2 = (float) 2;
```

```
var numero3 = 2f;
```

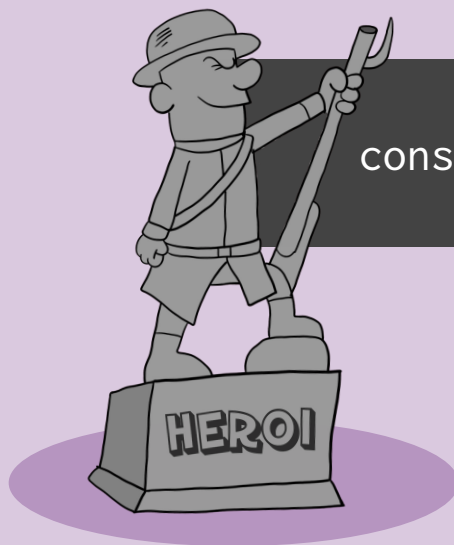
Jo faria un int, però
l'obligues a fer un
float



Les constants es defineixen posant **const** al davant.

No podrem modificar aquesta varibale més endavant!

*Han de ser avaluable
en temps de compilació*



```
const int estatua = 1
```

Operadors

Els booleans tenen els
operadors lògics

&&

i

||

o

!

no



És mamífer **&&** és un porc
Cert

és un porc **||** és un peix
Cert

! és un porc
Fals

és un porc **&&** és un peix
Fals

El resultat sempre serà
un altre booleà

```
bool esUnPorc = true;  
bool esUnMamifer = true;  
bool esUnPeix = false;
```

true

```
bool resultat = esUnPorc && esUnMamifer;
```

```
resultat = esUnPorc && esUnPeix;
```

false

```
resultat = esUnPorc || esUnPeix;
```

true

```
resultat = !esUnPorc;
```

false

Amb els dos **tipus numèrics** es poden fer **operacions matemàtiques**

```
int numero = 21;
```

Suma, resta

```
int doble = numero + numero;
```

```
int divuit = numero - 3;
```

Multiplicació i
divisió

```
int triple = numero * 3;
```

```
int meitat = numero / 2;
```

La divisió dona el
resultat sense
decimals si la
variable és int

```
int modul = numero % 2;
```

Retorna el mòdul (el
que sobra de la
divisió)

En qualsevol dels tipus
bàsics s'hi poden fer servir
operadors relacionals per
obtenir un valor booleà

Igual

==

Diferent

!=

Més gran

>

Més petit

<

>=

<=

El resultat dels
operadors relacionals
sempre és un booleà
(**true** o **false**)

```
int numero = 21;
```

```
bool resultat = numero == 21
```

true

```
string patata = "Patata";
```

```
resultat = patata != "Patata";
```

false

```
resultat = numero > 21;
```

false

```
resultat = numero <= 21;
```

true

```
bool x;
```

```
resultat = x == false;
```

true

Els tipus numèrics poden fer
servir els **operadors numèrics**

Mòdul

+

-

%

*

/

Increment

Decrement

++

--

Funcionen de la mateixa
forma que les
operacions numèriques

```
int numero = 21;  
numero = numero + 21
```

42

```
numero = numero - 20;
```

22

```
numero = numero * 2;
```

44

```
numero = numero / 4;
```

11

```
numero = numero % 5
```

1

```
numero++;
```

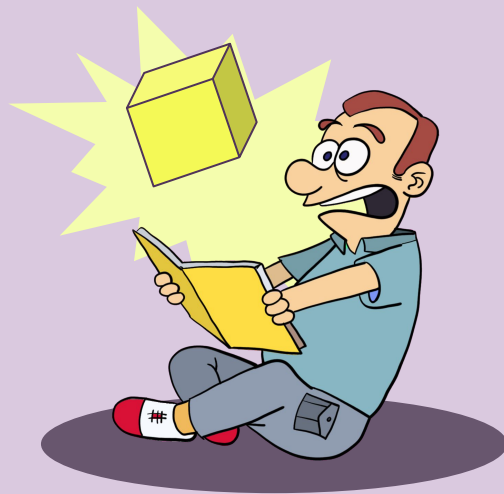
2

Dades compostes:
strings

Els **strings** són diferents
dels altres tipus de dades
bàsics



Són **objectes** i s'hi poden fer
operacions a partir dels
mètodes que ofereixen



Algunes de les
operacions que es poden
fer amb strings

```
string frase = " té més gana que seny";  
string nom = "En Manel";
```

```
nom.Length
```

8

```
nom.ToUpper()
```

EN MANEL

```
frase[8]
```

g

```
frase.IndexOf("y")
```

19

```
string.Concat(nom, frase);
```

En Manel té
més gana que
seny

```
nom + frase
```

En Manel té més
gana que seny

```
nom.Substring(3)
```

Manel

Els strings es poden
generar dinàmicament
amb **interpolació**
d'altres variables

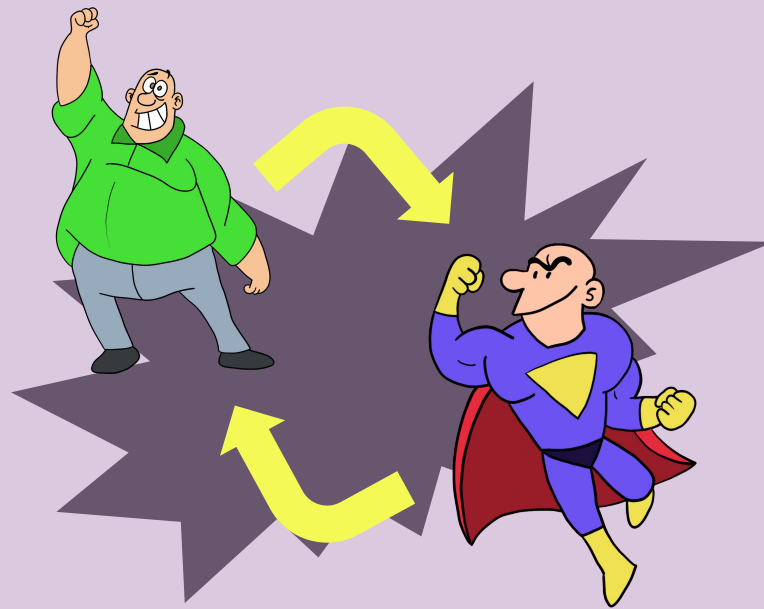
```
int edat = 20;  
string nom = "Frederic";  
altura = 1.5;
```

```
string saluda = $"Hola, em dic {nom}, tinc  
{edat} anys i faig {altura} metres";
```

"Hola, em dic Frederic, tinc 20 anys i
faig 1.5 metres"

**Conversió entre els
tipus**

Les variables d'un tipus **es**
poden convertir en altres
tipus



Convertir entre **tipus numèrics** és el més fàcil

```
int edat = 20;
```

```
// El podem convertir en float només assignant  
float edatAmbDecimals = edat;
```

```
// Un float el podem convertir en int
```

```
int edat2 = (int) edatAmbDecimals;
```

Li posem el tipus per evitar que surti un avís del fet que perdem dades (però funcionaria igual si no es posa)

Convertir els **números**
en **cadenes de caràcters**

```
int edat = 20;

// El podem convertir en string
string edatstring = edat.ToString();

// Per convertir un número amb decimals es fa igual
float tresImig = 3.5f
string tresImigString = tresImig.ToString();

// També es pot fer amb la interpolació
string edatstring2 = $"{edat}";
```

La conversió de **cadena**s de **caràcters** en **números** és una mica més problemàtica

```
string edat = "20";
```

```
// El podem convertir en int amb Parse  
int vintnumero = int.Parse(vint);  
Console.WriteLine($"Num: {vintnumero}");
```

```
// El sistema també funciona amb floats  
string tresImig = "3.5"  
float num = float.Parse(tresImig);  
Console.WriteLine($"float: {num}");
```

```
int x = int.Parse("Hola");
```

Si no pot fer el parse es genera una excepció

El **TryParse** retorna si la conversió ha anat bé sense generar excepcions

```
string edat = "20";  
  
var ok = int.TryParse(edat, out int numero);
```

Si **ok** és true,
ha anat bé

Si ha anat bé,
número tindrà
el valor

La conversió de cadenes a números **sempre s'ha de fer quan es llegeixen dades des del teclat** si cal el valor del número.

```
Console.Write("Entra un nombre: ");  
var edat = Console.ReadLine();
```

edat sempre
serà un string

```
// Si necessitem el número l'hem de convertir  
int numeroEdat = int.Parse(edat);
```


Versió 2 - 2025



INSTITUT
CENDRASSOS

Les imatges són meves o dels propietaris dels logos