

# Programació



UT1.2 Tipus de dades simples



# Manipulació de dades

- És important tindre clar el procés en el que es dóna **l'ordre a la memòria de guardar o recuperar dades**.
- **Dades:** tota la informació que l'ordinador utilitza en les execucions de programes.
- Cada dada es tracta per separat (Ex: programa que suma 2 nombres treballa amb tres dades)

# Tipus de dada

- Definició: conjunt de valors vàlids que pot prendre una dada i el conjunt d'operacions (transformacions) que se li pot aplicar
- Exemples de dades numèriques:
  - Una ciutat està a **8.25** km d'una altra
  - Una persona té **30** anys
  - Han passat **15** dies des d'un esdeveniment

# Tipus de dades

- Exemples dades alfabètiques:
  - El meu gat es diu **Milo**
  - Estudie a l'**IES Poeta Paco Mollá**
  - Visc al **Carrer del castell**
- Els valors que pot prendre cada dada són limitats (utilitzem memòria de l'ordinador, que és limitada)

# Tipus de dades

- Cada dada d'un programa **ha de pertànyer a algun tipus**
- És faena de programador **decidir quin tipus s'apropa més a la informació a representar**
- Cada llenguatge fa servir **tipus de dades bàsics (o primitius)**
- A partir d'estos tipus **es poden definir altres més complexos**

# Tipus de dades estàndard

- Són tipus que d'una manera o altra, tots els llenguatges suporten:
  - Enter
  - Real
  - Caràcter
  - Booleà

# Tipus de dada enter

- Representa un valor numèric, positiu o negatiu, sense decimals
- Exemples de literals(\*) enters: 3, 0, -345, ....
- Valor mínim i màxim: -2147483647 al 2147483647
- Exemples d'enters: edat, dia del mes, any, quantitat de fills, etc ...

(\*) Anomenem *LITERAL* a l'expressió d'un **valor concret** d'un tipus de dada.

# Exemple enter

```
public class LiteralsEnter {  
    public static void main (String[] args) {  
        System.out.println(3);  
        System.out.println(0);  
        System.out.println(-345);  
        System.out.println(138764);  
        System.out.println(-345002);  
    }  
}
```

*RECORDA: És important que executes aquests exemples, on es pot comprovar el funcionament dels literals de cada tipus de dada, i com els imprimeix per consola (println) el llenguatge de programació Java.*

# Exemple enter

En Java tenim quatre tipus de dades simples que representen valors enters: byte, enter curt (short), enter (integer) i enter llarg (long).

Si en Java volem forçar un literal per a que siga un enter llarg hem d'afegir una "L" al final del literal:

```
System.out.println(29); //Imprimeix un valor de tipus enter  
System.out.println(29L); //Imprimeix un valor de tipus enter llarg.
```

# Tipus de dada real

- Representa un valor numèric, positiu o negatiu, amb decimals.
- Exemples de literals: 2.25, 4.0, 45, 3., 100.3, .5, etc ...
- Exemples de dades: un preu en euros, el rècord mundial dels 100m llisos, la distància entre ciutats, etc ...

# Exemple Real

```
public class LiteralsReal {  
    public static void main (String[] args) {  
        System.out.println(2.25);  
        System.out.println(4.0);  
        System.out.println(-9653.3333);  
        System.out.println(100.0003);  
    }  
}
```

# Tipus Enter vs Real

- Per què existeix el tipus enter, si es pot representar el mateix valor amb tipus real?



# Exemple Real

En Java tenim dos tipus de dades que representen valors reals: flotants (float) i de doble precisió (double).

```
System.out.println(29);      // Imprimeix un literal de tipus enter
System.out.println(29.0);    // Imprimeix un literal de tipus double
System.out.println(29.);     // Imprimeix un literal de tipus double
System.out.println(29d);     // Imprimeix un literal de tipus double
System.out.println(29f);     // Imprimeix un literal de tipus float
System.out.println(29.f);    // Imprimeix un literal de tipus float
System.out.println(29.0f);   // Imprimeix un literal de tipus float
```

*Presta especial atenció a este tipus de dada i les seues operacions. Segurament estaràs menys habituat al seu ús*

# Tipus booleà

- Representa un valor de tipus lògic per **establir la veritat o falsedat** d'un estat o afirmació.
  - Només dos valors possibles: **true / false**
  - Exemples: interruptor (On / off), casat (Si / no), dret a vot, contrasenya és correcta, etc
- ...

# Exemple booleà

```
public class LiteralsBoolea {  
    public static void main (String[] args) {  
        System.out.println(true);  
        System.out.println(false);  
    }  
}
```

# Tipus de dada caràcter

- Representa una unitat fonamental de text usada en qualsevol alfabet, un nombre o signe de puntuació o exclamació.
- Exemples de literals: 'a', 'A', '4', '>', '?', 'Γ' (lletra grega gamma majúscula), etc ..
- Exemples de dades: cada un els símbols individuals de l'alfabet, etc ...

# Exemple Caràcter

```
public class LiteralsCaracter {  
    public static void main (String[] args) {  
        System.out.println('a');  
        System.out.println('A');  
        System.out.println('4');  
        System.out.println('>');  
        System.out.println('?');  
    }  
}
```

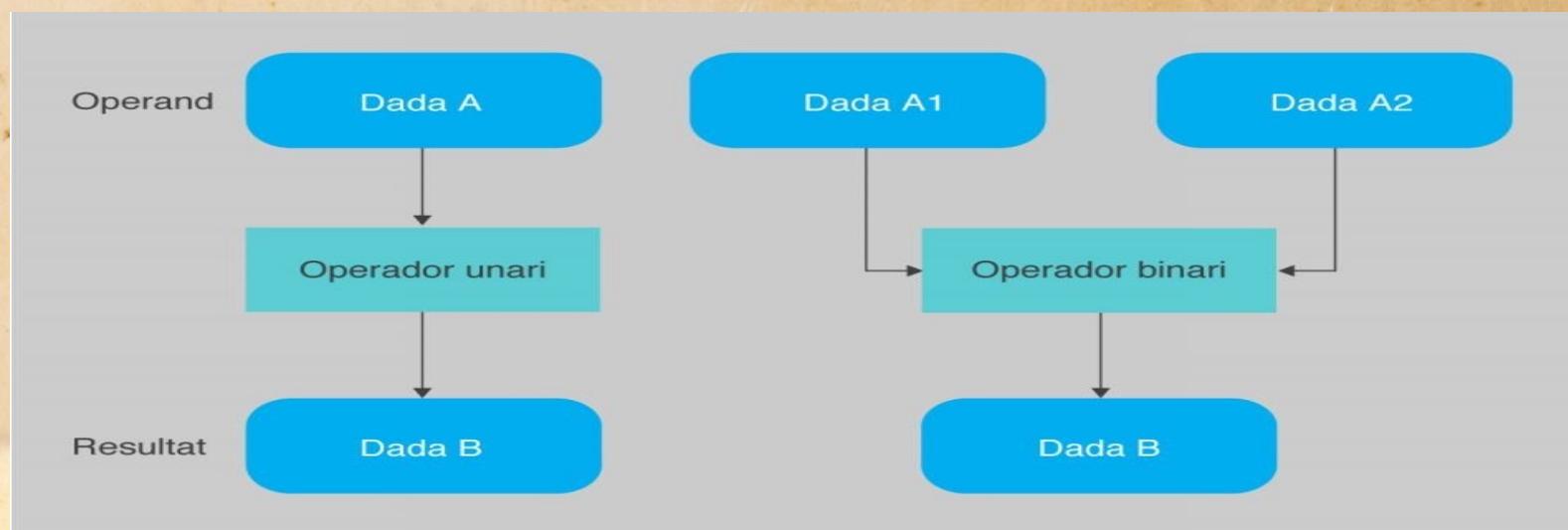
Tipo de datos	Información representada	Rango	Descripción
byte	Datos enteros	-128 ↔ +127	Se utilizan 8 bits (1 byte) para almacenar el dato.
short	Datos enteros	-32768 ↔ +32767	Dato de 16 bits de longitud (independientemente de la plataforma).
int	Datos enteros	-2147483648 ↔ +2147483647	Dato de 32 bits de longitud (independientemente de la plataforma).
long	Datos enteros	-9223372036854775808 ↔ +9223372036854775807	Dato de 64 bits de longitud (independientemente de la plataforma).
char	Datos enteros y caracteres	0 ↔ 65535	Este rango es para representar números en unicode, los ASCII se representan con los valores del 0 al 127. ASCII es un subconjunto del juego de caracteres Unicode.
float	Datos en coma flotante de 32 bits	Precisión aproximada de 7 dígitos	Dato en coma flotante de 32 bits en formato IEEE 754 (1 bit de signo, 8 para el exponente y 24 para la mantisa).
double	Datos en coma flotante de 64 bits	Precisión aproximada de 16 dígitos	Dato en coma flotante de 64 bits en formato IEEE 754 (1 bit de signo, 11 para el exponente y 52 para la mantisa).
boolean	Valores booleanos	true/false	Utilizado para evaluar si el resultado de una expresión booleanas es verdadero (true) o falso(false).

# Operació amb dades primitives

- Cada tipus de dada només es pot operar de maneres molt concretes
- **Operació:** acció per la qual un tipus de dada es transforma
- Cada tipus de dada admet únicament una sèrie d'operacions concretes
- **Exemple: no es pot sumar una dada de tipus enter amb una dada de tipus booleà**

# Operació amb dades primitives

- Aplicar operacions permet crear nous valors
- Tipus d'operacions:
  - **Unàries** (només un operand)
  - **Binàries** (dos operands)



# Operacions amb enters

- Aritmètics:
  - Unaris: - (Canvi de signe)
  - Binaris: +, -, \*, /, % (Qualsevol operació aritmètica entre dos dades de tipus enter donen com a resultat un altre enter)

```
// Operador suma
System.out.println(4 + 3);
// Operador multiplicació
System.out.println(4 * 3);
// Operació de mòdul
System.out.println(4 % 3);
// Operació de canvi de signe sobre l'operació anterior
System.out.println(-(4 % 3));
```

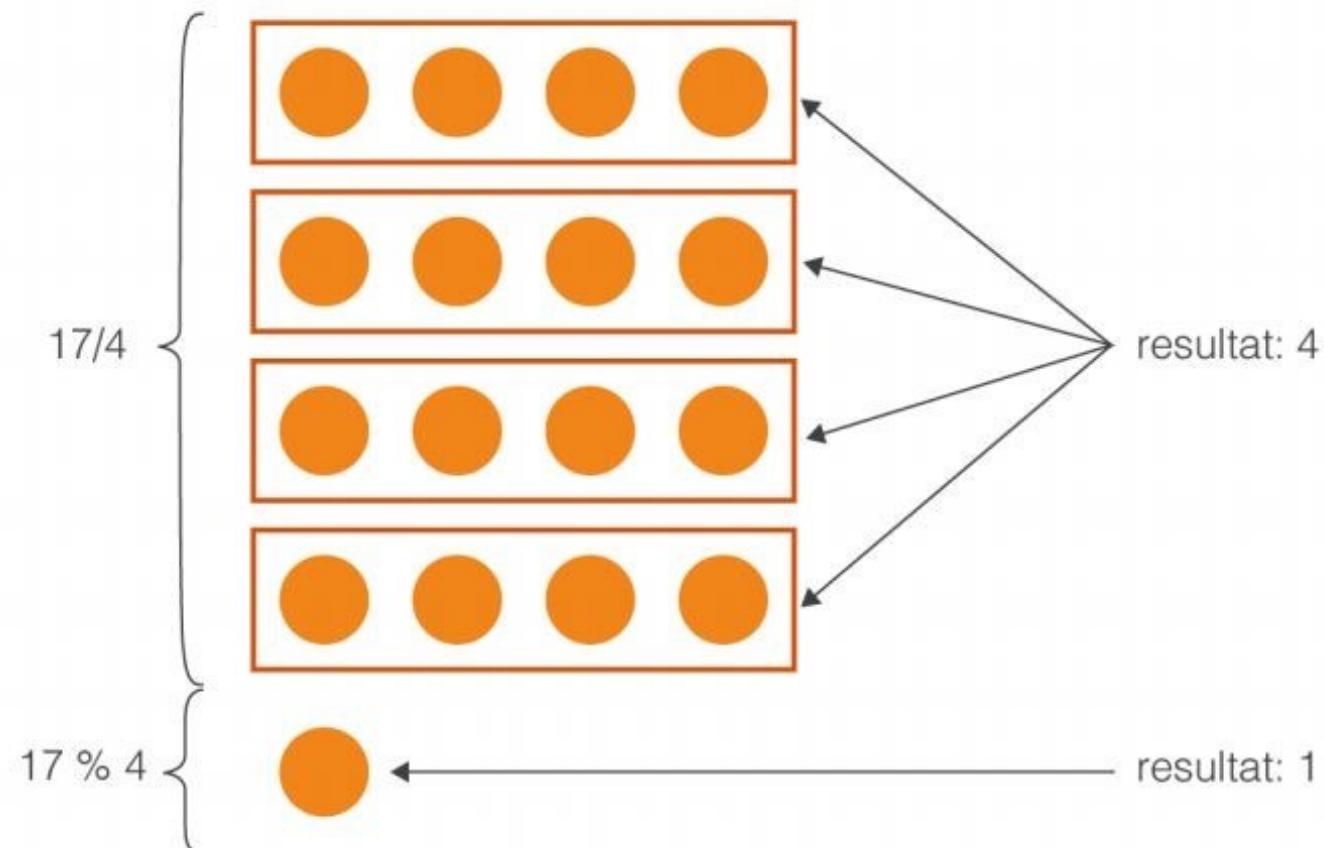
# Operadors divisió entera (/) i mòdul (%)

$$\begin{array}{r} 17 \longdiv{4} \\ \underline{1} ) \quad 4 \end{array}$$

Resultat de la divisió entera ( $17/4 = 4$ )

Resultat del residu o mòdul de la  
divisió entera ( $17 \% 4 = 1$ )

# Operadors divisió entera (/) i mòdul (%)



# Operadors amb enters

- Relacionals o de comparació:

- Binaris:

- == (igual)
    - != (different)
    - > (major que)
    - < (menor que)
    - >= (major o igual)
    - <= (menor o igual)

*Una operació relacional dona sempre com a resultat un valor de tipus booleà (un valor lògic), es a dir **true** o **false***

# Exemples op. relacionals

operands		Resultats de la operació				
A	B	A == B	A > B	A < B	A >= B	A ≤ B
4	3	false	true	false	true	false
14	-2	false	true	false	true	false
-78	34	false	false	true	false	true
12	12	true	false	false	true	true

# Pràctica 3



- Escriu el programa que representa els exemples de la diapositiva anterior

# Operadors amb enters

- De bits:
  - & (AND lògic bit a bit)
  - | (OR lògic bit a bit)
  - ^ (XOR lògic bit a bit)
  - << (desplaçament a l'esquerra de bits omplint amb zeros per la dreta)
  - >> (desplaçament a la dreta de bits omplint a l'esquerra amb el bit de signe)
  - >>> (desplaçament a la dreta de bits omplint amb zeros per l'esquerra)

# Operadors amb enters

exemples:

**12 | 10**

1100

1010

-----

**1110 (14)**

**12 & 10**

1100

1010

-----

**1000 (8)**

# Operacions amb reals

- Les mateixes que per als enters a excepció de les operacions de mòdul (%) i operadors de bits. Estes dos últimes operacions no apliquen per al tipus de dada real.

# Pràctica 4

- Escriu un programa que utilitze tots els operadors amb reals possibles fent ús de literals de reals diferents

# Operacions entre caràcters

- Només operacions relacionals:  
`'A' != 'b'`  
`'A' == 'A'`  
`'b' <= 'x'`  
`'T' > 'C'`

**Comprova que passa si fem servir  
l'operador + amb dos dades de tipus  
caràcter**

# Pràctica 5

- Escriu un programa que utilitze tots els operadors per a caràcters possibles fent ús de literals de caràcters diferents

# Operacions amb booleans

Operands		Resultats de l'operació		
A	B	A && B	A    B	! A
false	false	false	false	true
true	false	false	true	false
false	true	false	true	true
true	true	true	true	false

&& → conjunció (AND)  
|| → disjunció (OR)  
! → negació (NOT)

Investiga com funciona l'operador ^ (XOR) si tenim dos operands A i B de tipus booleà.

És important que quede ben clar el funcionament dels operadors booleans per a poder continuar avançant. Pots revisar el següent enllaç per a ampliar informació o pregunta al professor si tens qualsevol dubte:

<https://www.javatpoint.com/logical-operators>

# Operacions amb booleans

- Relacionals o de comparació:

operands	Resultats de la operació		
A	B	$A == B$	$A != B$
false	false	true	false
true	false	false	true
false	true	false	true
true	true	true	false

$== \rightarrow$  igual  
 $!= \rightarrow$  different

Suposeu que A i B són interruptors per comprovar la igualtat d'estat o no

**IMPORTANT:** Recorda que per a COMPARAR si dos valors són iguals utilitzem ==

# Exemple d'operacions amb booleans

```
public class OperacionsBoolea {  
    public static void main (String[] args) {  
        //Operacions lògiques: el resultat és un booleà  
        System.out.println(!true);  
        System.out.println(true && true);  
        System.out.println(true && false);  
        System.out.println(true || false);  
        System.out.println(false || false);  
        //Operacions relacionals: el resultat també és un booleà  
        System.out.println(false == false);  
        System.out.println(false != true);  
    }  
}
```

*RECORDA: És important que executes aquests exemples, on es pot comprovar el resultat de les operacions per a cada tipus de dada.*

# Construcció d' expressions

- **Expressió:** Combinació qualsevol d'operadors i operands .
- Ha de ser **correcta sintàctica i semàticament.**
- **Sintàcticament:**
  - Un literal per si mateix és una expressió
  - Donada una expressió  $E$ , és també correcta entre parèntesis  $(E)$ .
  - Donada una expressió  $E$  i un operador unari  $op$ ,  $Op\ E$  és una expressió correcta
  - Donades dos expressions  $E_1$  i  $E_2$  i un operador binari  $op$  ,  $E_1\ op\ E_2$  és una expressió correcta

# Expressions: exemples

- 6
- (6)
- -4
- $6 + 5$
- $(6 + 5) * -4$

# Construcció d' expressions



- **Semànticament**

- Qualsevol operació ha de ser entre dades del mateix tipus (o tipus compatibles)
- L'operació ha d'exsistir per al tipus de dada operat.

# Exemples erronis en Java

- `true + false`
- `-'g'`
- `5 == false`
- `5 || 4.0`
- `5 == '4'`
- `52 == '4'`

En estos casos el programa funciona, no dona cap error. ¿Per què penses que pot ser? ¿Què està fent eixa operació?

# Avaluació d'expressions

- Cal tindre en compte l'**ordre de precedència**. En cas d'empat, es resol sempre ordenadament d'esquerra a dreta

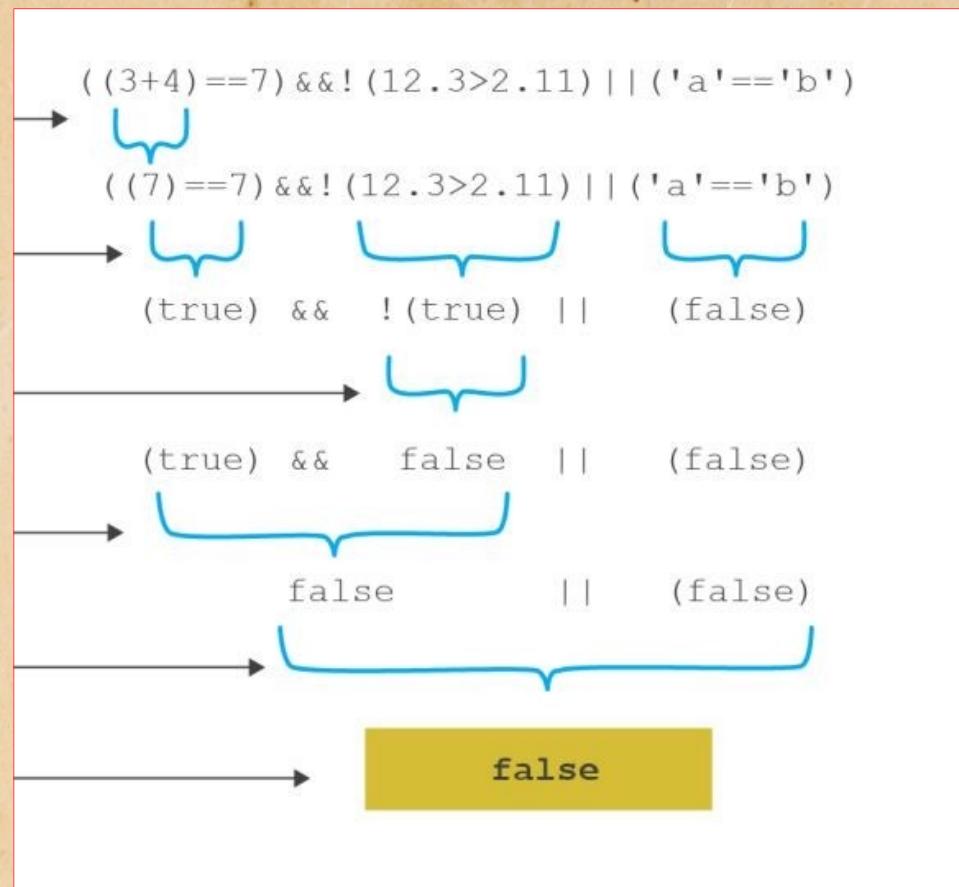
# Taula de prioritats



ordre	operació	operador
1	Canvi de signe	- (unari)
2	Negació	! (unari)
3	Producte, divisió i mòdul	* / %
4	Suma i resta	+ -
5	Relacionals de comparació	> < <= >=
6	Relacionals d'igualtat	== !=
7	Conjunció	&&
8	Disjunció	

# Exemple

$((3 + 4) == 7) \&\& !(12.3 > 2.11) \mid\mid ('a' == 'B')$



# Pràctica 6

- Fes un programa que continga una expressió amb literals dels quatre tipus de dades vistos i ho mostre per pantalla.

El programa ha de gestionar almenys 10 literals.

# Desbordament i errors de precisió

Quan executem este programa apareix un error. Comprova-ho.

```
//Un programa que usa un enter mooooolt gran
public class TresMilMillions {
    public static void main (String[] args) {
        System.out.println(3000000000);
    }
}
```

Exception in thread "main" java.lang.RuntimeException: Uncompilable code - integer number too large

# Desbordament i errors de precisió

- El problema és que hem escollit un tipus de dada inadequat. Això és un error comú del programador.
- Per a poder emmagatzemar eixe valor hem de fer ús del tipus de dada “long” o enter llarg

```
System.out.println(3000000000L);
```