UD3.- Elements del llenguatge Java

Mòdul: Programació
1r DAM
Curs 2018-2019

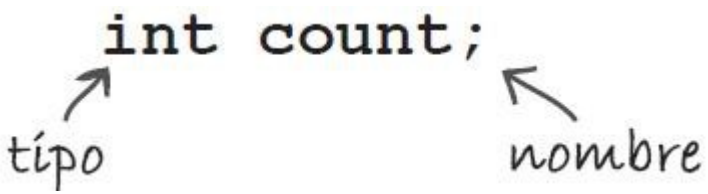
Unión Europea
Fondo Social Europeo
El FSE invierte en tu futuro

INDEX

- Variables
- Tipus de dades
- Constants
- Literals
- Operadors

Variables

- Permeten emmagatzemar dades, resultats i valors intermedis d'un programa
- Tenen associat un tipus de dades, que determina:
  - Conjunt de valors que la variable pot guardar
  - Operacions que es poden realitzar amb eixa variable
- Declaració de variables en Java:

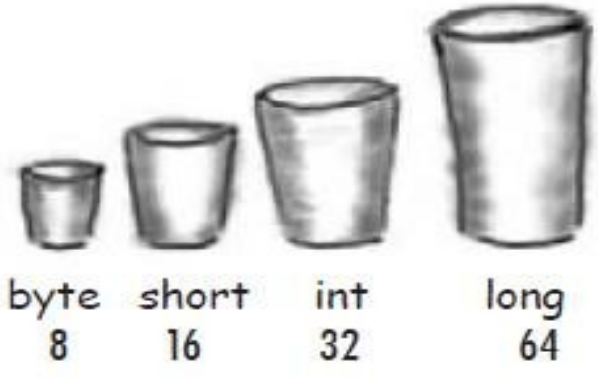


Tipus de dades en Java

- Tipus de dades primitius
  - boolean, byte, char, double, float, int, long, short
- Tipus de dades referència
  - String, Array, Class (objectes), Interface

Tipus de dades primitius

Tipus numèrics



TIPUS ENTER

TIPUS DECIMAL

NOM	GRANDÀRIA EN BITS	VALOR MÀXIM
byte	8	127
short	16	32767
int	32	2147483647
long	64	9223372036854775807
float	32	3.4E+38
double	64	1.7E+308

Tipus de dades primitius

- Tipus caràcter

NOM	GRANDÀRIA EN BITS	REPRESENTA
char	16	Representa caràcteres (lletres, números i símbols especials) Java utilitza la codificació UNICODE de 2 bytes

- Tipus booleà o lògic

NOM	GRANDÀRIA EN BITS	REPRESENTA
boolean	8	Pot prendre els valors *true* i *false*

Tipus de dades primitius. Exemples

Declaración y asignación de primitivas

```
int x;
x = 234;
byte b = 89;
boolean isFun = true;
double d = 3456.98;
char c = 'f';
int z = x;
```

```
boolean isPunkRock;
isPunkRock = false;
boolean powerOn;
powerOn = isFun;
long big = 3456789;
float f = 32.5f;
```

Para indicar que se trata de un float, porque Java si encuentra un punto decimal, se piensa que es un doble, a no ser que encuentre la 'f'.

Què passa si el valor és major?

- No podem posar molta aigua en un got xicotet.
- Sí podem, però se n'ix del got.
- Per exemple:
  - `int x = 24;`
  - `byte b = x;`
  - `// no funciona!!`
- El valor d'x no serà exacte
- Es perdran bits i precisió

Assignació de variables

- Assignant un valor literal després del símbol =
  - `x=12;`
  - `isGood=true;`
- Assignant el valor d'una variable a una altra
  - `x=y;`
- Utilitzant una expressió combinació de les dos anteriors
  - `x=y+43;`
- Assignant el valor en la mateixa sentència de declaració
  - `char car='A', car2='\u0041';`
  - `float precioPatata=1.2F, precioChoco=2.3F;`

Àmbit de les variables

- En Java n'hi ha tres tipus de variables:
  - Variables d'instància: són aquelles que tindran tots els objectes que es crearan d'una classe. Tindran, en termes generals, valors diferents per a cada objecte.
  - Variables de classe: seran iguals a tots els objectes de la classe.
  - Variables locals: són les que s'utilitzen en els mètodes.

Àmbit de les variables(II)

- L'àmbit d'una variable és la part del programa en la qual és coneguda i es pot utilitzar.
- Una variable local es declara dins del cos d'un mètode d'una classe i és visible únicament dins d'eixe mètode.
- Les variables es poden declarar en qualsevol lloc del cos del mètode, fins i tot després d'instruccions executables, encara que és una bona costum declarar-les al principi.
- També poden declarar-se variables dins d'un bloc entre {…}
  - Únicament seran "visibles" dins d'eixe bloc.
  - Les variables definides en un bloc han de tindre noms diferents.

Identificadors de les variables

- Els identificadors són els noms que se'ls donen a les variables, classes, interfícies, atributs i mètodes d'un programa.
- Quins noms poden tindre les variables?
  - Ha de començar amb una lletra, subratllat (_) o el caràcter $. No pot començar amb un número.
  - Després del primer caràcter, sí que es poden utilitzar números. No es poden utilitzar espais en blanc ni els símbols dels operadors.
  - No poden coincidir amb cap paraula reservada.
  - El % no està permés, sí el $ i la ç. També accents i ñ, però no són recomanables.
  - Recorda que Java és *case sensitive*.

Llistat de paraules reservades en Java

boolean	byte	char	double	float	int	long	short	public	private
protected	abstract	final	native	static	strictfp	synchronized	transient	volatile	if
else	do	while	switch	case	default	for	break	continue	assert
class	extends	implements	import	instanceof	interface	new	package	super	this
catch	finally	try	throw	throws	return	void	const	goto	enum

I els valors *true* i *false*

Convencions per als identificadors

- Els noms de les variables i els mètodes haurien de començar per minúscula i els de les classes per majúscula.
- Si l'identificador està format per vàries paraules, la primera s'escriu en minúscules (excepte per a les classes) i la resta de paraules comencen per majúscula.
  - variable: anyDeCreacio
  - classe: HolaMundo

Constants

- Per a declarar una constant utilitzem el modificador *final*
  - final double PI = 3.1415926536;
- El valor d'una constant no pot ser modificat al llarg d'un programa, a diferència de les variables.
- Hem de donar-li un valor quan el declarem.

Assignació de variables. Exemples

`int size = 32;`	declara un entero llamado ***size*** asignándole el valor 32
`char initial = 'j';`	declara un carácter llamado ***initial*** asignándole el valor 'j'
`double d = 456.709;`	declara un doble llamado ***d*** asignándole el valor 456.709
`boolean isCrazy;`	declara un boolean llamado ***isCrazy*** sin asignarle valor
`isCrazy = true;`	asigna el valor true a la variable ***isCrazy***, antes definida
`int y = x + 456;`	declara un entero llamado ***y*** y le asigna el valor de la suma del valor de x más el valor 456

Literals

- Un literal és un valor que s'expressa a ell mateix.

- Un literal enter pot expressar-se:
  - en decimal (base 10) → Exemple:21
  - en octal (base 8) → Exemple: 025
  - en hexadecimal (base 16) → Exemple: x03A
  - pot afegir-se al final la lletra L o l per indicar que l'enter és considerat com un long.

Literals

- Un literal real o decimal pot expressar-se:
  - part entera, (.) i part fraccionaria
    - Exemples: 345.678, 0.00056
  - notació exponencial o científica
    - Exemples: 3.45678e2, 5.6e-4
  - per defecte és de tipus double. Si volem que s'interprete com a *float* haurem d'afegir el sufixe F o f.

Literals

- Literal carácter: es representen entre cometes simples (‘). Pot ser:
  - Un símbol (lletra)
    - Exeples: ‘a’ , ‘B’ , ‘{‘ , ‘ñ’ , ‘á’
  - El codi Unicode del caràcter en octal o hexadecimal.
    - ’\141’ → codi Unicode en octal per a ’a’
    - ’\u0061’ → codi Unicode en hexadecimal per a ’a’
- Una “seqüència d'escape”, per a caràcters especials

Seqüència	Significat
'\"'	Cometa simple
'\”'	Cometa doble
'\\'	Contrabarra
'\b'	Backspace
'\n'	Canvi de línia
'\r'	Retorn de carro
'\t'	Tabulador

Literals

- Literal booleà
  - paraules reservades *true* i *false*
    - Exemple: boolean activado=false;
- Literal *string* o cadena de caràcters
  - No formen part dels tipus de dades elementals de Java
  - Entre cometes dobles (“)
    - Exemples:
      - System.out.println(”Primera línea\nSegunda línea del string\n”);
      - System.out.println(”Hol\u0061”);

Conversió de tipus de dades

- Quan es realitza una assignació:
  - identificador = expressió;
- tant la variable com l'expressió han de ser del mateix tipus o de tipus compatibles

- Una expressió pot assignar-se a una variable sempre que siga d'un tipus de menor grandària que el tipus de la variable. Per tant, podem assignar en el següent ordre:
  - byte → short → int → long → float → double

Conversió de tipus de dades

- Altres formes de conversió de tipus es poden realitzar explícitament a través del *casting*.
  - (tipus) expressió
    - Ejemplo: num= (int) 34.56
- També utilitzant funcions adequades d'alguns paquets de Java.

Operadors. Operadors unaris

- Signe
  - Posar un signe + o un signe – davant d'una expressió.
  - Exemples: +45, -32

Operadors. Operadors unaris

- Increment (++) i decrement (--)
  - Augmentar i disminuir en 1 el valor de la variable
  - Poden anar davant (pre) o darrere (post) de la variable.
  - Exemple:
    - `int valor, i=5;`
    - `i++;  //ara i val 6; és equivalent a i=i+1`
    - `i--;    //ara i val 5; és equivalent a i=i-1`
  - La diferència entre pre i post apareix en una instrucció complexa:
    - `valor=i++; //ara valor val 5, i val 6`
      `// equival a { valor=i; i=i+1; }`
    - `valor=++i; // ara valor val 7, i val 7`
      `// equival a { i=i+1; valor=i; }`

Operadors. Operadors aritmètics

- Poden realitzar operacions aritmètiques que impliquen el càlcul de valors numèrics representats per literals, variables, altres expressions, crides a funciones i propietats i constants.

- Sintaxis

```
expresion1 operador_aritmético expresion2
```

- Exemple

```
int x;
x = 52 * 17;
x = 120 / 4;
x = 67 + 34;
x = 32 - 12;
X = 7 % 2;
```

Operadors. Operadors aritmètics

- + suma
- - resta
- * multiplicació
- / divisió
- % resto de la divisió entera

Operadors. Operadors de comparació

- Símbols que avaluen expressions condicionals i retornen un *boolean*.

- Sintaxis

```
expresion1 operador_de_comparación expresion2
```

Operadors. Operadors de comparació

Operador	*true* si...	*false* si...
< (menor que)	expr1<expr2	expr1>=expr2
<= (menor o igual que)	expr1<=expr2	expr1>expr2
> (major que)	expr1>expr2	expr1<=expr2
>= (major o igual que)	expr1>=expr2	expr1<expr2
= = (igual)	expr1==expr2	expr1!=expr2
!= (distint)	expr1!=expr2	expr1==expr2

Operadors. Operadors de comparació

- Exemple

```
int cantidad;
boolean pedidoGrande;
pedidoGrande = cantidad > 1000;
```

```
boolean testResult ;
testResult = ( 45 < 35 );
testResult = ( 45 == 45 );
testResult = (4 != 3 );
testResult = ('a' > 'b');
```

Mètode equals()

- El mètode equals() s'utilitza per comparar els valors dels objectes.
- Un dels usos més comuns es per a comparar dos cadenes String (tipus no primitiu).

```
String s = new String("Hola");
if(s.equals("Hola")) {
   System.out.println("Son iguales");
}
```

Mètode equals() vs operador ==

- L'operador == s'utilitza per comparar valors de tipus primitius de dades, es a dir, tornarà true quan els dos valors siguen iguals.
- Si apliquem l'operador == a objectes (variables de referència a objectes) el que comparará es si les referències apunten al mateix objecte, es a dir, apunten a la mateixa adreça de memòria.
- Per tant no hem d'utilitzar el operador == per comparar objectes (ni tipus Strings o qualsevol dada que no siga un tipus primitiu).

Mètode equals() vs operador ==

```
String cadena = new String("Hola");

String missatge = new String("Hola");

//Seran diferents perquè són objectes distints
if(cadena == missatge) {
  System.out.println("Són iguals");
} else {
  System.out.println("Són diferents");
}
//Seran iguals perquè es compara el seu valor
if(cadena.equals(missatge)) {
  System.out.println("Són iguals");
} else {
  System.out.println("Són diferents");
}
```

Operadors. Operadors lògics

- Realitzen una avaluació lògica d'expressions i retornen un valor *boolean*.
- Sintaxis

```
expresion1 operador_lógico expresion2
```

- Exemple

```
edad>18 && sexo=='H'

edad>18 || sexo=='H'
```

Operadors. Operadors lògics

Operador	Funció
&&	Combina dos expressions. Cada expressió ha de ser *true* per a que tota l'expressió siga *true*.
||	Combina dos expressions. Si una expressió és *true,* tota l'expressió és *true*.
!	Dona el negatiu lògic de l'entrada.

Operadors lògics. Taules de veritat

Operador AND

expr1	expr2	resultat
false	false	false
false	true	false
true	false	false
true	true	true

Operador NOT

expr1	resultat
false	true
true	false

expr1	expr2	resultat
false	false	false
false	true	true
true	false	true
true	true	true

Operador OR

Combinar operadors lògics i de comparació

- Podem combinar operadors de comparació y operadors lògics amb instruccions condicionals.

```
edad <= 25 && edad >=14
```

Operadors. Operadors d'assignació

Operador	Exemple
=	edad=34;
+=	edad+=1; edad=edad+1;
-=	edad-=3; edad=edad-3;
*=	edad*=2; edad=edad*2;
/=	edad/=2; edad=edad/2;
%=	edad%=2; edad=edad%2;

Operadors. Operadors de concatenació

- Permet generar una cadena de caràcters a partir d'altres dos

  expr1 + expr2

- Exemple: “Hola” + “,” + “bon día”

Operadors. Prioritat

- 1.Paréntesi, de dins cap a fora.
- 2. Unaris
- 3. Aritmètics
  - A. Multiplicativos: * / %
  - B. Sumativos: + -
- 4. Comparatius o relacionales
- 5. Lògics o booleans
  - A. NOT
  - B. AND
  - C. OR
- 6. Assignació
- Quan apareixen operadors de la mateixa prioritat junts en una expressió, el compilador avalúa cada operació d'esquerre a dreta.

Operadors. Prioritat

- Exemple:
  - a = -3 + 5 + 2 * 4 – 4 / 2 * 3 - 5 % 2;
  - Quin és el valor final d'esta expressió?
  - I de la següent expressió?
  - b = -3 + 5 + 2 * 4 – 6 / 4 * 3 - 5 % 2;

Visualització d'informació per pantalla

- Mètodes *print()* i *println()*
  - *println()* inclou un salt de línia al final de l'eixida

```
System.out.print("Se imprime este mensaje sin el
salto de línea");

System.out.println("Se imprime este mensaje con
un salto de línea");
```

Entrada de dades per teclat

- El mètode *read( )* llig un únic caràcter

```
char c = (char) System.in.read();
```

- Esta forma de llegir la informació del teclat és molt poc pràctica (imagina que haguérem de llegir cada línia de text caràcter a caràcter...)
  - Solució: declarar un objecte de la classe *Scanner* i emprar els seus mètodes.

Entrada de dades per teclat

```
//Importem el fitxer on està la classe
import java.util.Scanner;

public class Exemple {
	public static void main (String[] args) {
		int primerNum;
		//Es declara l'objecte lector de la classe Scanner
		Scanner lector = new Scanner(System.in);
			...
		System.out.print("Escriu un número i polsa retorn: ");

		//Es llig un valor enter per teclat i s'espera al retorn.
		primerNum = lector.nextInt();
		lector.nextLine();
	...
	}
}
```

Entrada de dades per teclat. Mètodes

Mètode	Tipus de dades llegit
lector.nextByte()	byte
lector.nextShort()	short
lector.nextInt()	int
lector.nextLong()	long
lector.nextFloat()	float
lector.nextDouble()	double
lector.nextBoolean()	boolean
lector.next()	String
lector.nextLine()	String

Tipus enumerats

- Són conjunts de valors constants per als quals no n'hi ha un tipus predefinit.
  - Per exemple: per a representar els dies de la setmana, estacions de l'any, mesos de l'any, etc...

```
public class Semana {

    public enum DiaSemana{LUNES, MARTES, MIERCOLES,
                          JUEVES, VIERNES, SABADO, DOMINGO}

    public static void main (String[] args){

        DiaSemana hoy = DiaSemana.JUEVES;

        DiaSemana ultimo=DiaSemana.DOMINGO;

        System.out.println( "Hoy es " +hoy +"\n Y el ultimo dia es "
                            +ultimo );
    }
}
```