

M03 - UF4 - POO

Introducció a JAVA



Manuel Ruiz Morante
Curs 2019-20

[INTRODUCCIÓ A JAVA](#)

[COM FUNCIONA?](#)

[Oracle JDK vs OpenJDK](#)

[INSTAL·LACIÓ JDK](#)

[OracleJDK](#)

[OpenJDK](#)

[PRIMER PROGRAMA \(sense IDE\)](#)

[INSTAL·LACIÓ IDE](#)

[Eclipse IDE for Java Developers](#)

[ELEMENTS DEL LLENGUATGE](#)

[Variables](#)

[Declaració](#)

[Visibilitat \(Scope\)](#)

[Constants](#)

[Casting](#)

[String i StringBuilder](#)

[Arrays](#)

[Convencions nom de les variables](#)

[Operadors](#)

[Aritmètics](#)

[Assignació](#)

[Incrementals](#)

[Lògics](#)

[Relacionals](#)

[Condicional \(?\)](#)

[Instanceof](#)

[Estructures de programació](#)

[Comentaris](#)

[Javadoc](#)

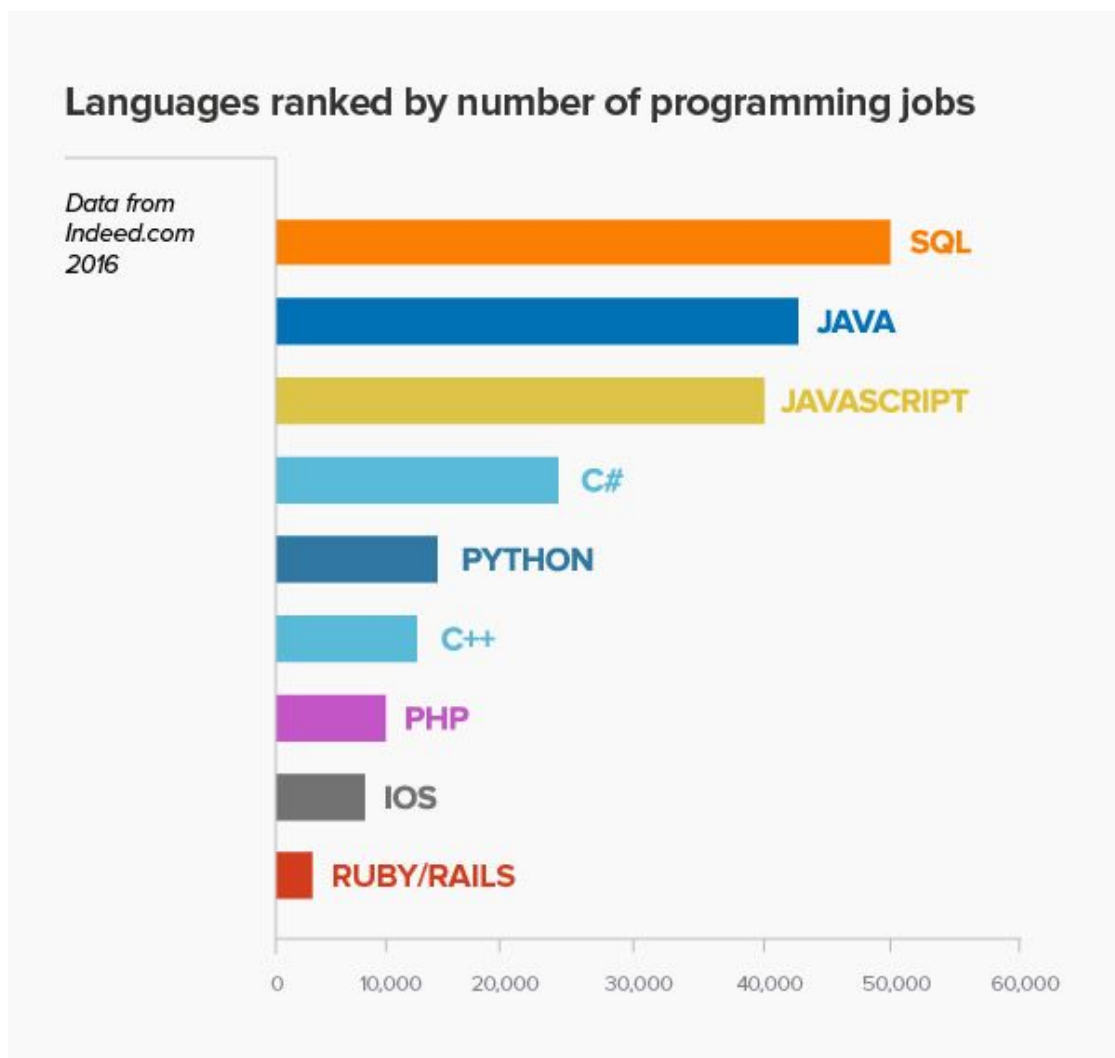
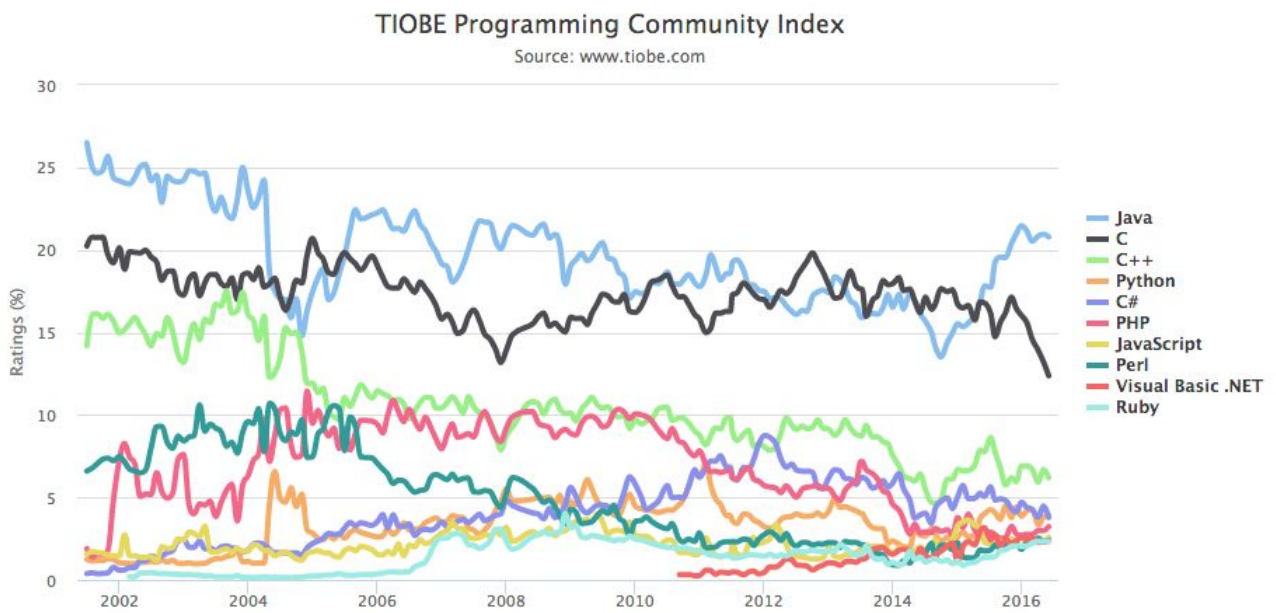
[Bifurcacions](#)

[Bucles](#)

[Entrada de dades per teclat](#)

[DEPURACIÓ D'ERRORS a Eclipse](#)

1. INTRODUCCIÓ A JAVA



[Video animat evolució ranking a Stack Overflow](#)

Plataformas Java

Java Card

Java SE

Java EE

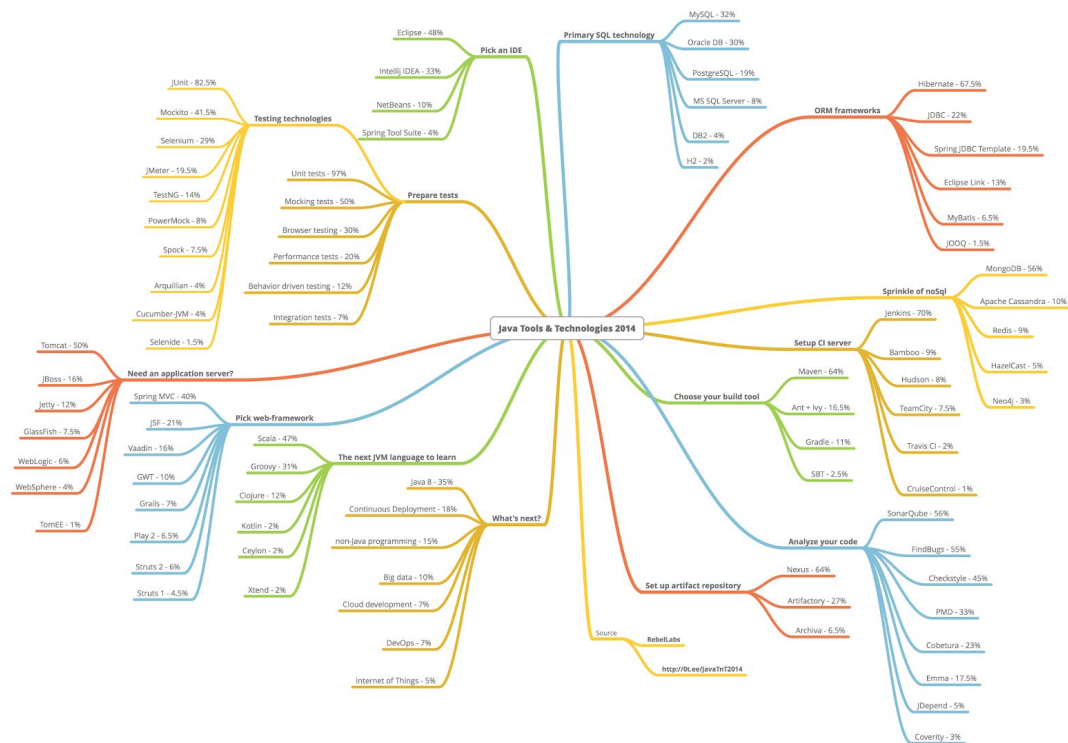
Java ME

Java FX

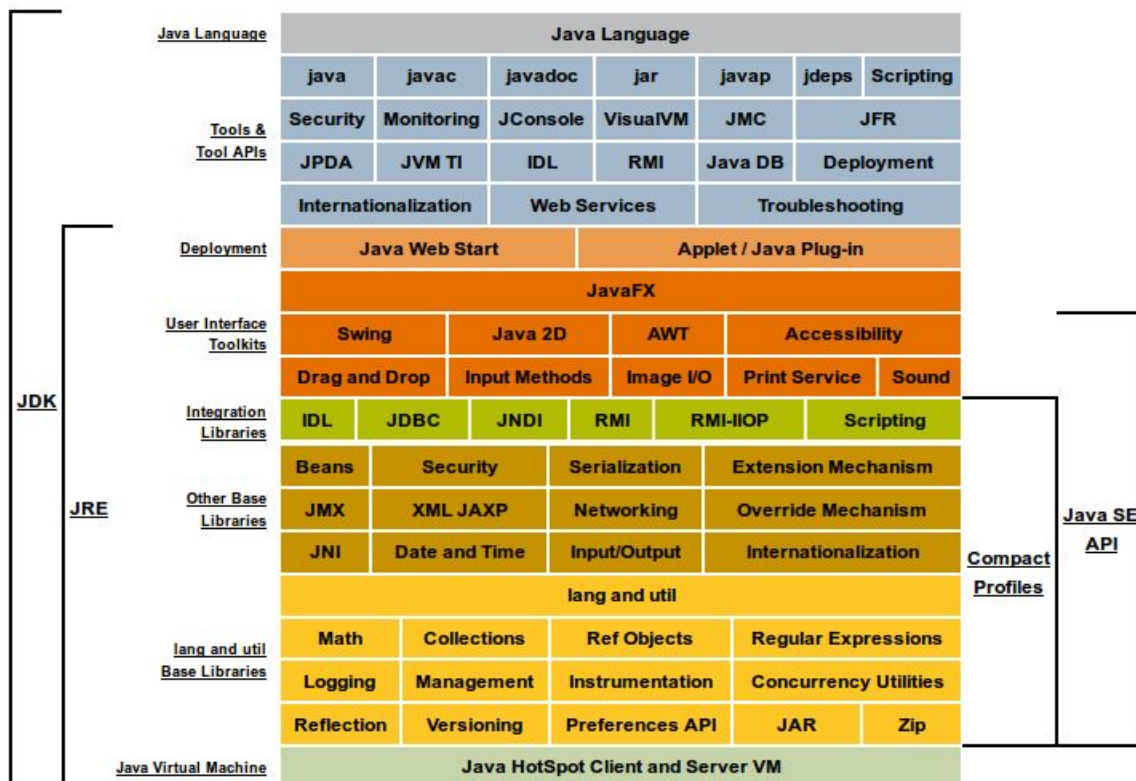


+

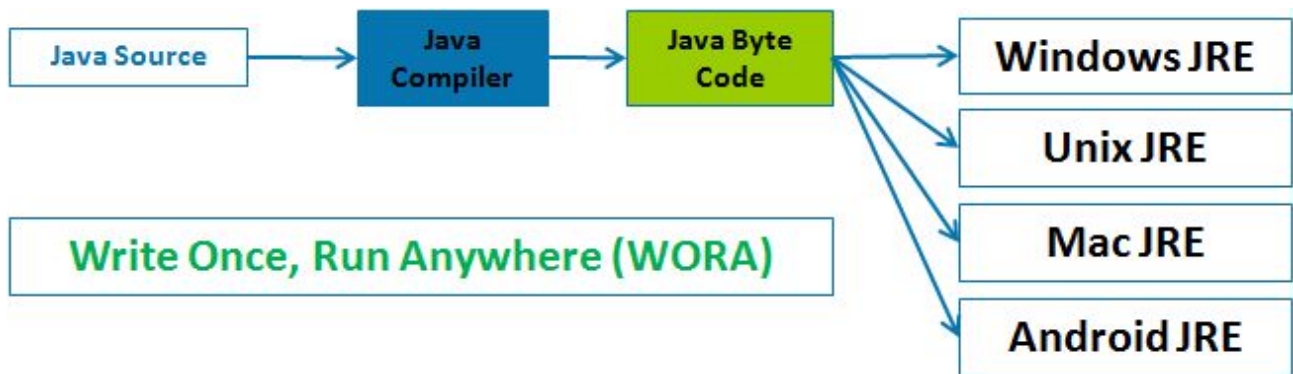




Java SE



2. COM FUNCIONA?



3. Oracle JDK vs OpenJDK

A partir del 16 d'Abril de 2019 Oracle, propietària actual de JAVA, ha canviat la llicència de JAVA, i ha passat a ser de pagament per poder executar aplicacions d'ús empresarial. Només permet l'ús sense cost per *personal use* i *development use*.

Important Oracle JDK License Update

The Oracle JDK License has changed for releases starting April 16, 2019.

The new [Oracle Technology Network License Agreement for Oracle Java SE](#) is substantially different from prior Oracle JDK licenses. The new license permits certain uses, such as personal use and development use, at no cost – but other uses authorized under prior Oracle JDK licenses may no longer be available. Please review the terms carefully before downloading and using this product. An FAQ is available [here](#).

Commercial license and support is available with a low cost [Java SE Subscription](#).

Oracle also provides the latest OpenJDK release under the open source [GPL License](#) at jdk.java.net.

És per això que moltes empreses s'estan [passant a utilitzar OpenJDK](#), el desenvolupament open source on hi col·laboren moltes empreses, Oracle inclosa.

Existeixen algunes [diferències](#) i no es pot dir [quina és millor que l'altra](#) sinó que dependrà de l'ús i el suport que vulguem, així que per nosaltres seran bàsicament el mateix, l'únic problema és que per descarregar la OracleJDK hem de tenir compte a Oracle.

<https://www.digitalocean.com/community/tutorials/instalar-java-en-ubuntu-con-apt-get-es>

4. INSTAL·LACIÓ JDK

Utilitzarem la darrera versió LTS, que és la v11. L'anterior LTS és la v8. Poden coexistir diferents versions a la vegada.

OracleJDK

[Windows, Linux, Mac](#)

OpenJDK

[Linux, des de repositoris:](#)

```
sudo apt-get install default-jdk
```

[Windows, Mac, etc](#)

[Finalment, cal apuntar la variable d'entorn JAVA_HOME al directori d'instal·lació.](#)

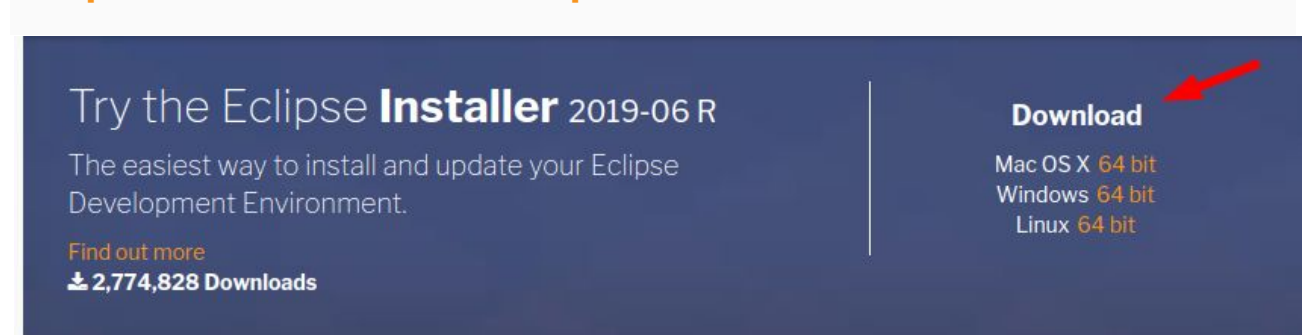
5. PRIMER PROGRAMA (sense IDE)

- public static void main (String[] args)
- Compilació amb javac
- Execució amb java

6. INSTAL·LACIÓ IDE

L'eina de desenvolupament que utilitzarem al mòdul és Eclipse (for JAVA Developers), que podeu descarregar des d'aquí:

[Eclipse IDE for Java Developers](#)



Si ja teniu alguna versió de Eclipse instal·lada no cal instal·lar una nova.

També podeu utilitzar altres entorns de desenvolupament com: Netbeans, Visual Studio Code, IntelliJ IDEA, etc.

7. ELEMENTS DEL LLENGUATGE

7.1. Variables

Tipus de dades

		NOMBRE	TIPO	OCUPA	RANGO APROXIMADO
TIPOS DE DATOS EN JAVA	TIPOS PRIMITIVOS (sin métodos; no son objetos; no necesitan una invocación para ser creados)	byte	Entero	1 byte	-128 a 127
		short	Entero	2 bytes	-32768 a 32767
		int	Entero	4 bytes	$2 \cdot 10^9$
		long	Entero	8 bytes	Muy grande
		float	Decimal simple	4 bytes	Muy grande
		double	Decimal doble	8 bytes	Muy grande
		char	Carácter simple	2 bytes	---
		boolean	Valor true o false	1 byte	---
	TIPOS OBJETO (con métodos, necesitan una invocación para ser creados)	Tipos de la biblioteca estándar de Java	String (cadenas de texto) Muchos otros (p.ej. Scanner, TreeSet, ArrayList...)		
		Tipos definidos por el programador / usuario	Cualquiera que se nos ocurra, por ejemplo Taxi, Autobus, Tranvia		
		arrays	Serie de elementos o formación tipo vector o matriz. Lo consideraremos un objeto especial que carece de métodos.		
		Tipos envoltorio o wrapper (Equivalentes a los tipos primitivos pero como objetos.)	Byte		
			Short		
			Integer		
			Long		
			Float		
			Double		
			Character		
			Boolean		

Declaració

tipus_primitiu nomVariable = valor_inicial;

int i = 0;

Classe nomObjecte = null;

```
Classe nomObjecte = new Classe();  
    String cadena = new String();  
    Persona juan = new Persona();
```

Visibilitat (Scope)

Les variables són visibles dintre de les claus { } on han estat declarades.

Constants

```
final double PI = 3.1415;
```

Casting

```
double d = 67.8;  
int i = (int) d; //La variable origen ha de "càpigner" dintre de la variable destí.
```

També podem utilitzar les classes de tipus Wrapper (Integer, Long, etc) i la classe String:

```
int i = Integer.parseInt(cadena);
```

```
String s = String.valueOf(num);
```

String i StringBuilder

Una cadena és una seqüència de caràcters (lletres, números, caràcters especials, etc).

A JAVA, una cadena és un objecte de la classe [String](#), inclosa al paquet **java.lang**

```
String cadena1 = "Hola";  
String cadena2 = new String ("Adéu");
```

Aquestes dues opcions són [gairabé iguals](#).

Un String no es pot modificar. Tot i que si utilitzem el concatenador "+" així ho sembli, en realitat s'està creant un nou String. Si el codi requereix modificar el contingut d'una cadena, hem d'utilitzar la classe [StringBuilder](#).

StringBuilder	String
Changeable	Immutable
Easier insertion, deletion, and replacement.	Easier concatenation.
Can be more difficult to use, especially when using regular expressions (introduced in the next lesson).	Visually simpler to use, similar to primitive types rather than objects.
Use when memory needs to be conserved.	Use with simpler programs where memory is not a concern.

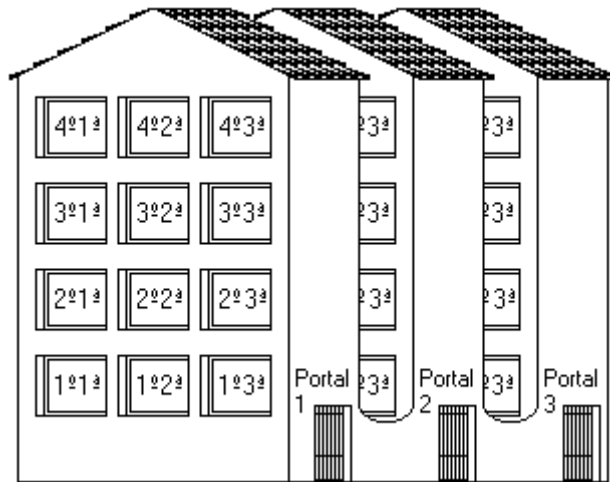
JP_7_1_sg

Arrays

És un tipus especial d'objecte.

Es poden definir arrays de 1,2,3 dimensions:





```
tipus_dada[] nomArray = new tipus_dada[capacitat_i];
tipus_dada[][] nomArray = new tipus_dada[capacitat_i][capacitat_j];
```



- El primer element és el 0.
- Capacitat màxima coneguda.
- `nomArray.lenght` ens dóna la longitud del array.

Convencions nom de les variables



- Case sensitive
- Comencen amb minúscula. Si està formada per dos paraules, la segona comença en majúscula
 - `dataFin`
- Ha d'identificar el contingut
- Constants senceres en majúscules

7.2. Operadors

Aritmètics

`+`, `-`, `*`, `/`, `%` (residu)

Assignació

Operador	Utilització	Expresión equivalente
<code>+=</code>	<code>op1 += op2</code>	<code>op1 = op1 + op2</code>
<code>-=</code>	<code>op1 -= op2</code>	<code>op1 = op1 - op2</code>
<code>*=</code>	<code>op1 *= op2</code>	<code>op1 = op1 * op2</code>
<code>/=</code>	<code>op1 /= op2</code>	<code>op1 = op1 / op2</code>
<code>%=</code>	<code>op1 %= op2</code>	<code>op1 = op1 % op2</code>

Incrementals

`++` `a++` `a = a + 1`
`--` `a--` `a = a - 1`

Lògics

Operador	Nombre	Utilització	Resultado
<code>&&</code>	AND	<code>op1 && op2</code>	true si op1 y op2 son true. Si op1 es false ya no se evalúa op2
<code> </code>	OR	<code>op1 op2</code>	true si op1 u op2 son true. Si op1 es true ya no se evalúa op2
<code>!</code>	negación	<code>! op</code>	true si op es false y false si op es true
<code>&</code>	AND	<code>op1 & op2</code>	true si op1 y op2 son true. Siempre se evalúa op2
<code> </code>	OR	<code>op1 op2</code>	true si op1 u op2 son true. Siempre se evalúa op2

Relacionals

Operador	Utilització	El resultado es true
<code>></code>	<code>op1 > op2</code>	si op1 es mayor que op2
<code>>=</code>	<code>op1 >= op2</code>	si op1 es mayor o igual que op2
<code><</code>	<code>op1 < op2</code>	si op1 es menor que op2
<code><=</code>	<code>op1 <= op2</code>	si op1 es menor o igual que op2
<code>==</code>	<code>op1 == op2</code>	si op1 y op2 son iguales
<code>!=</code>	<code>op1 != op2</code>	si op1 y op2 son diferentes



== , != no es pot utilitzar per comparar el contingut de dos Strings! ja que en aquest cas estem comparant la posició de la memòria i no el contingut, que s'ha de comparar amb el mètode equals()

Condicional (?)

Funciona com un "if" però en una sola línia:

`variable = booleanExpression ? res1 : res2`

Instanceof

objectName **instanceof** ClassName

true → L'objecte és de la classe

false → L'objecte NO és d'aquesta classe

7.3. Estructures de programació

Comentaris

//línia comentada

/*

conjunt de línies comentades

..

*/


Javadoc

/**

Documentació del nostre codi per generar el javadoc

*/

Bifurcacions

```
if (condició){   
    instruccions;  
}
```

```
if (condició) {  
    instruccions;  
} else {  
    instruccions2;  
}
```

```
if (condició1) {  
    instruccions1;  
} else if (condició2) {  
    instruccions2;  
} else if (condició3) {  
    instruccions3;  
} else {  
    instruccions4;  
}
```

switch (variable) {

```
    case valor1: instrucciones1; break;
    case valor2: instrucciones2; break;
    case valor3: instrucciones3; break;
    case valor4: instrucciones4; break;
    [default: instrucciones5;]
}
```



Bucles

```
int i=0;
```

```
while (i<10) {
```

```
    instrucciones;
    instrucciones;
    instrucciones;
    instrucciones;
    instrucciones;
    instrucciones;
    instrucciones;
    instrucciones;
```

```
    i++;
```

```
}
```

```
do {
```

```
    instrucciones;
```

```
} while (condició);
```



```
for (inicialització; condició permanència al bucle; increment) {
```

```
    instrucciones;
```

```
}
```

```
String[] diasSemana = new String(7);
```

```
for (int i=0;i<diasSemana.length();i++){
```

```
    System.out.println(diasSemana[i]);
```

```
}
```



ArrayIndexOutOfBoundsException

```
//foreach
```

```
for (tipus_dada element: llista) {
```



```

        instrucciones;
    }

    for (String diaSemana: diasSemana) {
        System.out.println(diaSemana);
    }

```

8. Entrada de dades per teclat

Existeixen dues opcions per fer la recollida de dades de l'usuari per teclat o entrada estàndard (System.in):

- [java.util.Scanner](#)
Scanner sc = new Scanner(System.in);
- [java.io.BufferedReader](#)
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

[Exemple d'ús](#)

9. DEPURACIÓ D'ERRORS a Eclipse

Una de les eines més útils per trobar errors a les aplicacions és mitjançant el concepte de debugg.

Es tracta d'executar el codi pas a pas per veure quines línies de codi s'executen en cada moment i quin valor tenen les variables que fem servir.

Per debuggar amb **Eclipse** farem servir bàsicament les següents funcionalitats:

- **Breakpoint** → Línia del codi on es parará el debugger per començar a revisar el codi
- **F5 - Step into** → Executar una línia de codi, si aquesta línia de codi és una funció, entrarà dintre d'aquesta per començar-la a executar
- **F6 - Step over** → Executar una línia de codi, si aquesta línia de codi és una funció, no entrarà dintre d'aquesta i la executarà sencera
- **F8 - Resume** → Continuar la execució del programa fins trobar un altra breakpoint o finalitzar la execució del tot
- **CTRL + SHIFT + i** → Veure ràpidament el valor d'una variable