

Cadenes alfanumèriques i conversions de format

Cadenes de caràcters (String)

[Apunts de Strings en Java](#)

[Classe String](#)

[Apunts d'Expresions regulars en Java](#)

Exemples amb String

```
Scanner lector = new Scanner(System.in);
//declarar i instanciar String (són objectes, cal invocar el constructor)
String nom = new String("Lluís");
//instanciació i inicialització abreujada amb constant
String salutacio = "Hola";
//ús de l'operador de concatenació
String missatge = salutacio + " " + nom;
System.out.println(missatge);
//obtenir la longitud del string
System.out.println("La longitud del missatge és "+missatge.length());
//obtenir el caràcter en una posició
System.out.print("Quin índex? ");
int index = lector.nextInt();
try {
    char c = missatge.charAt(index); //pot llançar
    StringIndexOutOfBoundsException si l'índex està fora de límits
    System.out.println("El caràcter a la posició "+index+" és "+ c);
} catch (StringIndexOutOfBoundsException e) {
    System.out.println("Índex incorrecte");
}
//comparar strings (negatiu, zero o positiu segons el resultat de la comparació)
int comp = "Hola".compareTo("Holo");
System.out.println(comp);
//ús del mètode concat() per concatenar String
System.out.println(salutacio.concat(nom));
//mètodes per analitzar el contingut
String frase = "En un lugar de la Mancha de cuyo nombre no quiero acordarme";
System.out.println("Comença per En? "+ frase.startsWith("En"));
System.out.println("Acaba per rme? "+ frase.endsWith("rme"));
//igualtat de strings
String a = "Taula";
String b = "taula";
System.out.println("Son iguals? "+ a.equals(b));
System.out.println("Son iguals (ignorant case)? "+ a.equalsIgnoreCase(b));
//ús del mètode format() per obtenir un string amb dades formatades
int edat = 22;
double salari = 1800.0;
String informacio =
    String.format("%s tens %d anys i salari %.2f\n",
```

```
        nom, edat, salari);
System.out.println(informacio);
//trobar la posició d'un caràcter o string (indexOf(), lastIndexOf())
System.out.println("La primera 'u' és a l'índex: "+frase.indexOf('u'));
System.out.println("La darrera 'u' és a l'índex: "+frase.lastIndexOf('u'));
System.out.println("La primera 'de' és a l'índex: "+frase.indexOf("de"));
//ús del mètode replace()
System.out.println(frase.replace('e', '3'));
//extracció de fragments del string (mètodes substring())
System.out.println(frase.substring(4, 20));
//conversió a majúsc/minusc
System.out.println(frase.toUpperCase());
System.out.println(frase.toLowerCase());
```

La sortida del codi anterior és:

```
Hola Lluís
La longitud del missatge és 10
Quin índex? 3
El caràcter a la posició 3 és a
-14
HolaLluís
Comença per En? true
Acaba per rme? true
Son iguals? false
Son iguals (ignorant case)? true
Lluís tens 22 anys i salari 1800,00

La primera 'u' és a l'índex: 3
La darrera 'u' és a l'índex: 44
La primera 'de' és a l'índex: 12
En un lugar d3 la Mancha d3 cuyo nombr3 no qui3ro acordarm3
n lugar de la Ma
EN UN LUGAR DE LA MANCHA DE CUYO NOMBRE NO QUIERO ACORDARME
en un lugar de la mancha de cuyo nombre no quiero acordarme
```

Lectura de string amb espais i lectura de línies amb Scanner

Quan cal llegir amb la classe [Scanner](#) una línia sencera en un String amb independència de si conté espais i altres separadors equivalents a l'espai (tabuladors, canvis de línia, etc.) es pot utilitzar el mètode **Scanner.nextLine()**.

Si es vol separar una entrada d'una línia en diverses lectures, fins i tot amb tipus de dada diferents, cal canviar el delimitador per defecte que fa servir Scanner. Per fer-ho, utilitzem el mètode **Scanner.userDelimiter()**.

```
Scanner useDelimiter(String pattern)
Scanner useDelimiter(Pattern pattern)
```

Exemple:

```
import java.util.Scanner;
/**
 * Read variables in a line with terminador '\n' and field separator ';'
 * @author Jose
 */
```

```
public class ScannerCsv {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        //show default delimiter
        System.out.println("Scanner default delimiter: "+sc.delimiter());
        //newlines and semicolons with any amount of spaces around are allowed
        sc.useDelimiter("(\\s*;\\s*)|\\n");
        //read a name (String with spaces allowed) and an integer age
        System.out.print("Input name and age: ");
        String name = sc.next();    //read name
        int age = sc.nextInt();    //read age
        //show variables read
        System.out.format("name: %s; age: %d\\n", name, age);
    }
}
```

Exemples

Usar un delimitador per separar un text en blocs i imprimir cada bloc en una línia.

```
import java.util.Scanner;

/**
 * Prints vertically a text splitting in tokens using a delimiter
 * @author Jose
 */
public class SplitText {

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        scan.useDelimiter("\\n"); //read until new line
        //input text
        System.out.print("Input a text: ");
        String text = scan.next();
        //remove leading and trailing whitespaces
        text = text.strip();
        //String text = scan.nextLine();
        System.out.println("Text: "+ text);
        //input character to use as a delimiter to split text
        System.out.print("Input a character: ");
        char c = scan.next().charAt(0);
        //char c = ' '; //delimiter to search
        //iterative solution
        System.out.println("Iterative solution");
        printSplittedTextIterative(text, c);
        //recursive solution
        System.out.println("Recursive solution");
        printFirstToken(text, c);
        //solution using Scanner
        System.out.println("Solution using Scanner");
        printSplittedTextUsingScanner(text, c);
    }

    /**
     * Prints text vertically using a delimiter (iterative solucion)
     * @param text the text to print
     */
}
```

```
* @param delimiter the delimiter to be used to split text
*/
public static void printSplittedTextIterative(String text, char delimiter) {
    int index = 0;
    int previousIndex = -1;
    String token;
    while ((index != -1) && (index < text.length())) {
        index = text.indexOf(delimiter, previousIndex+1);
        if (index < 0) { //delimiter not found
            token = text.substring(previousIndex+1);
        } else { //delimiter found
            token = text.substring(previousIndex+1, index);
            previousIndex = index;
        }
        if (token.length() > 0) System.out.println(token);
    }
}

/**
 * Prints text vertically using a delimiter (recursive solution)
 * @param text the text to print
 * @param delimiter the delimiter to be used to split text
 */
public static void printFirstToken(String text, char delimiter) {
    String token;
    int index = text.indexOf(delimiter);
    if (index < 0) { //delimiter not found, base case
        //final case
        token = text.substring(0);
        if (token.length() > 0) System.out.println(token);
    } else { //delimiter found, recursive case
        token = text.substring(0, index);
        if (token.length() > 0) System.out.println(token);
        printFirstToken(text.substring(index+1), delimiter);
    }
}

/**
 * Prints text vertically using a delimiter (using Scanner)
 * @param text the text to print
 * @param delimiter the delimiter to be used to split text
 */
public static void printSplittedTextUsingScanner(String text, char
delimiter) {
    Scanner sc = new Scanner(text);
    sc.useDelimiter(String.valueOf(delimiter));
    while (sc.hasNext()) {
        String token = sc.next();
        if (token.length() > 0) System.out.println(token);
    }
}
}
```

Conversió entre String i Number

Java proveeix tota una sèria de classes envolupants (*wrappers*) dels tipus de dades primitives (int, long, ...).

Totes aquestes classes són subclasses de la classe [*Number*](#):

- Integer
- Short
- Long
- Float
- Double etc.

Per convertir números a format string i viceversa, utilitzem aquestes classes envolupants i la classe String.

```
//donat un text que conté un número
String intText = "23";
//s'obté el valor numèric amb el mètode estàtic parseXXX de la classe envolupant
del tipus primitiu corresponent
int intValue = Integer.parseInt(intText);

//donat un valor numèric
int intValue = 23;
//s'obté la representació en format String amb el mètode toString de la classe
envolupant del tipus primitiu corresponent
String intText = Integer.toString(intValue);
//alternativament, la classe String proveeix els mètodes valueOf, els quals fan
el mateix que toString de la classe envolupant
String intText = String.valueOf(intValue);
```

Proposta d'exercici: El xifrat per desplaçament (xifrat Cèsar)

El xifrat utilitzat per Juli Cèsar per comunicar-se sense que els missatges poguessin ser llegits per l'enemic aplicava l'algorisme de xifrat per desplaçament.

Consulteu el seu funcionament [aquí](#).

Codifiqueu un programa que demani a l'usuari un missatge (String) i un desplaçament (enter), codifiqui el missatge, el mostri codificat, després el decodifiqui i el mostri decodificat.

Els mètodes per codificar i decodificar estaran en una classe separada (sense mètode main): XifratCesar.java.

```
/**
 * xifra el missatge amb l'algorisme de desplaçament
 * @param missatge el missatge a xifrar
 * @param desp el desplaçament a aplicar
 * @return el missatge xifrat
 */
public static String xifrarCesar(String missatge, int desp) {
    String result="";
    //TODO
    return result;
}
```

```
}  
  
/**  
 * desxifra el missatge amb l'algorisme de desplaçament  
 * @param missatge el missatge a desxifrar  
 * @param desp el desplaçament a aplicar  
 * @return el missatge desxifrat  
 */  
public static String desxifrarCesar(String missatge, int desp) {  
    String result="";  
    //TODO  
    return result;  
}
```

La classe principal importarà la classe XifratCesar i usará els seus mètodes per xifrar i desxifrar.

```
public class XifratDeMissatges {  
  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        sc.useDelimiter("\n");  
        System.out.print("Entra la frase a xifrar: ");  
        String textAXifrar = sc.next();  
        System.out.print("Entra el desplaçament: ");  
        int desplaçament = sc.nextInt();  
        //  
        System.out.println("Text a xifrar: "+ textAXifrar);  
        //  
        String textXifrat = XifratCesar.xifrarCesar(textAXifrar, desplaçament);  
        System.out.println("Text xifrat: "+ textXifrat);  
        //  
        String textDesxifrat = XifratCesar.desxifrarCesar(textXifrat,  
desplaçament);  
        System.out.println("Text desxifrat: "+ textDesxifrat);  
  
    }  
}
```

Solució:

- [Xifrat César](#)
- [Xifrat de missatges \(principal\)](#)

Dates i temps

Convé utilitzar les classes del paquet [java.time](#), les quals estan basades en el calendari ISO, el qual segueix les regles del calendari Gregorià introduït l'any 1582.

Les classes més destacades del paquet *java.time* són:

- **LocalDate**: representa dates sense hora i permet declarar-les, sumar-les, restar-les i comparar-les.
- **LocalTime**: l'equivalent a l'anterior, però per representar hores sense data.
- **LocalDateTime**: combinació de les dues anteriors, representa data i hora.

- **Instant:** emmagatzema un moment determinat en el temps amb data i hora com un timestamp.
- **ZonedDateTime:** Com LocalDateTime però tenint en consideració la zona horària.
- **Period:** representa diferències entre moments en el temps.
- **Duration:** similar a l'anterior, però només per a hores.

Totes aquestes classes produeixen instàncies immutables i no tenen constructors públics, sinó que es construeixen a partir de mètodes *factory*.

Els mètodes més importants són:

- **now():** crea instàncies noves a partir de la data i hora actuals.
- **of():** construeix dates i hores a partir de les seves parts.
- **with():** modifica la data o hora actual segons el paràmetre.

Per a obtenir les parts d'una data o d'una hora disposen de mètodes **getHour()**, **getMinute()**, **getMonth()**, etc.

```
//get current data
LocalDate today = LocalDate.now();
System.out.println(today);
//output: 2022-11-05
```

```
//get current time
LocalTime now = LocalTime.now();
System.out.println(now);
//output: 18:35:38.330782700
```

```
//obtenir la data i l'hora actuals
LocalDateTime todayNow = LocalDateTime.now();
System.out.println(todayNow);
//output: 2022-11-05T18:35:38.330782700
//use a formatter to convert into string with specific format
DateTimeFormatter dtFormat1 = DateTimeFormatter.ofPattern("EEEE yyyy/MMMM/dd, hh:mm:ss");
System.out.println(todayNow.format(dtFormat1));
//output: sábado 2022/noviembre/05, 06:35:38
```

```
//get current date and time with time zone
ZonedDateTime todayNowHere = ZonedDateTime.now();
DateTimeFormatter dtFormat2 = DateTimeFormatter
    .ofLocalizedDateTime(FormatStyle.FULL) //format
    .withLocale(Locale.forLanguageTag("ca-ES")); //locale
System.out.println(todayNowHere.format(dtFormat2));
//output: dissabte, 5 de novembre de 2022, a les 18:35:38 (Hora estàndard del
Centre d'Europa)
//especify format with language tag
DateTimeFormatter dtFormat3 = DateTimeFormatter.ofPattern("EEEE yyyy/MMMM/dd, hh:mm:ss")
    .withLocale(Locale.forLanguageTag("ca-ES"));
System.out.println(todayNowHere.format(dtFormat3));
//output: dissabte 2022/de novembre/05, 06:35:38
//especify format with DateTimeFormatter constants
System.out.println(todayNowHere.format(DateTimeFormatter.ISO_DATE_TIME));
//output: 2022-11-05T18:35:38.363622+01:00[Europe/Madrid]
```

```
//determine is a year is leap year
int year = 2024, month=3, day=5;
LocalDate date = LocalDate.of(year, month, day);
System.out.format("Is %d leap year?: %s ", date.getYear(),
date.isLeapYear()? "yes": "no");
```

Per a analitzar (*parse*) dates disposen del mètode *parse()*.

```
LocalDate hoy = LocalDate.parse("2023-10-25");
LocalDate seisNov = LocalDate.parse("25/10/2023",
DateTimeFormatter.ofPattern("d/M/yyyy") );
```

Per a més exemples: [Cómo manejar correctamente fechas en Java: el paquete java.time](#)

La classe StringBuilder

Classe StringBuilder

```
/**
 * Exemple d'ús de StringBuilder
 * @author Jose
 */
public class StringBuilderExemple {

    public static void main(String[] args) {
        //instanciar un StringBuilder amb un String
        StringBuilder sb = new StringBuilder("En un lugar de la Mancha");
        //convertir a String i mostrar
        System.out.println("=contingut inicial=");
        System.out.println(sb.toString());
        //append
        System.out.println("=append=");
        sb.append(" de cuyo nombre no quiero acordarme");
        System.out.println(sb.toString());
        //insert
        System.out.println("=insert=");
        sb.insert(12, "(esto es de El Quijote) ");
        System.out.println(sb.toString());
        //delete
        System.out.println("=delete=");
        int index = sb.indexOf("la");
        sb.delete(index, index+13);
        System.out.println(sb.toString());
        //demostració d'encadenament d'accions
        System.out.println("=encadenament d'accions=");
        sb.append("aaaa").append("bbbb").insert(5, "cccc");
        System.out.println(sb.toString());
        //reverse
        System.out.println("=reverse=");
        sb.reverse();
        System.out.println(sb.toString());
    }
}
```

Sortida del codi d'exemple anterior:


```
=contingut inicial=  
En un lugar de la Mancha  
=append=  
En un lugar de la Mancha de cuyo nombre no quiero acordarme  
=insert=  
En un lugar (esto es de El Quijote) de la Mancha de cuyo nombre no quiero  
acordarme  
=delete=  
En un lugar (esto es de El Quijote) de cuyo nombre no quiero acordarme  
=encadenament d'accions=  
En uncccc lugar (esto es de El Quijote) de cuyo nombre no quiero  
acordarmeaaaaabbbb  
=reverse=  
bbbbaaaaemradroca oreiuq on erbmon oyuc ed )etojiuQ lE ed se otse( ragul cccnu  
nE
```

Estructures unidimensionals



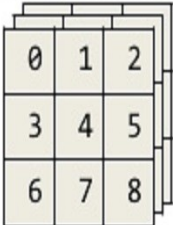
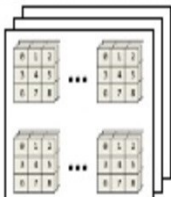
Arrays en Java

Un *array* és un grup de variables del mateix tipus referides amb un nom comú.

Són objectes, és a dir, són de tipus referencial. Això implica que s'emmagatzemen de forma dinàmica i cal instanciar-los. Els elements de l'array s'emmagatzemen en memòria de forma consecutiva.

Poden ser unidimensionals o de més d'una dimensió.

What is an array?

Dimensions	Example	Terminology
1		Vector
2		Matrix
3		3D Array (3 rd order Tensor)
N		ND Array

Com a objectes, tenen atributs i mètodes.

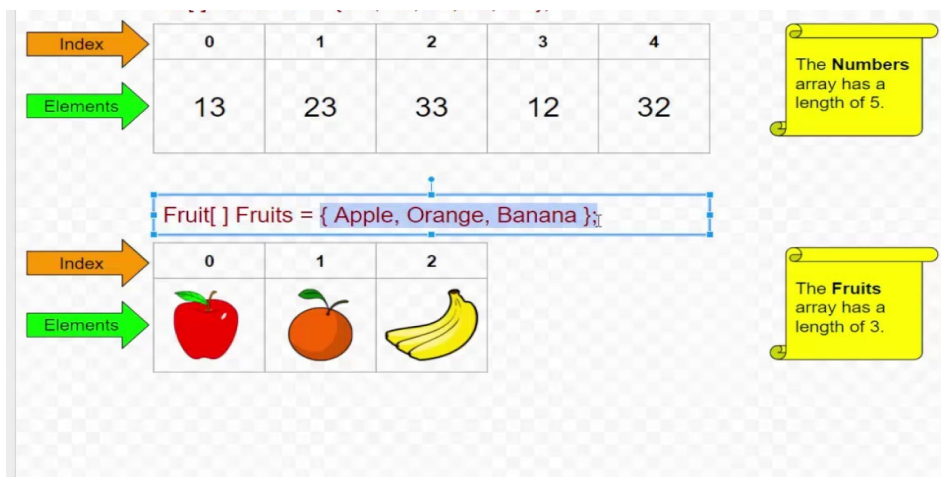
En aquest apartat ens ocuparem dels arrays unidimensionals.

Els elements de l'array s'organitzen seqüencialment. Cada element té en anterior (llevat del primer, amb índex 0) i un posterior (llevat del darrer).

La seva longitud es pot llegir del seu atribut **length**.

L'accés als seus elements es fa de manera directa utilitzant l'índex de la posició.

Un array pot contenir dades de qualsevol tipus primitiu o referencial, incloent-hi altres arrays.



Declaració d'arrays

La declaració d'un array declara l'identificador i el tipus d'element que conté.

```
type ident[];
type [] ident;
```

Exemples

```
int intArray[];
int [] intArray[];
float [] floatArray;
String [] stringArray;
```

La declaració no reserva espai en memòria per als elements de l'array. Com que l'array és un objecte, cal instanciar-lo definint a la vegada el nombre d'elements a emmagatzemar.

Instanciació d'arrays

```
ident = new type[size];
```

Exemples:

```
int [] intArray; //declaració
intArray = new int[10]; //instanciació de l'array de 10 elements int.
```

Es pot combinar la declaració i la instanciació:

```
int [] intArray = new int[10];
```

Definició abreujada d'arrays amb constants

Es pot declarar, instanciar i inicialitzar els elements d'un array amb la següent notació abreujada:

```
int [] intArray = {10, 20, 30, 40, 50}; //array de 5 elements int
```

Accés als elements d'un array

L'accés als elements d'un array es fa de manera indexada amb els operadors []. Cal tenir en compte que el primer element té índex **0** i el darrer **length-1**.

```
int [] intArray = {10, 20, 30, 40, 50}; //array de 5 elements int
int valor = intArray[3]; //accés a l'element amb índex 3
System.out.println(valor); //mostra 40
intArray[2]=31; //canvia el valor de l'element amb índex 2 (el 30 ara és 31)
```

Podem utilitzar un bucle per recórrer els elements de l'array

```
for (int i=0; i<intArray.length; i++) {
    System.out.println("Element amb índex "+i+": "+intArray[i]);
}
```

ArrayIndexOutOfBoundsException

Quan s'intenta accedir a un elements fora de l'array, es genera l'excepció

ArrayIndexOutOfBoundsException.

```
int intArray = new int[5];
//...
intArray[7] //llança excepció
```

Arrays d'objectes

Quan els elements de l'array són objectes, la instanciació de l'array només crea espai per a les referències als objectes.

Abans de llegir un element, cal instanciar l'objecte que conté.

```
String [] names = new String[3]; //declaració de l'array de 3 elements String,
els quals encara no s'ha creat (són nuls)
System.out.println(names[0]); //NullPointerException
```

```
String [] names = {"John", "Martha", "Louis"}; //declara, instancia i
inicialitza
System.out.println(names[0]); //mostra "John"
```

```
String [] names = new String[3]; //declaració de l'array de 3 elements String,
els quals encara no s'ha creat (són nuls)
names[0] = "John"; //instancia i inicialitza String i copia referència la
primera posició de l'array
names[1] = "Martha";
names[2] = "Louis";
System.out.println(names[1]); //mostra "Martha"
```

La sintaxi general per a qualsevol tipus d'objecte (substituir String pel nom de la classe):

```
String [] names = new String[3]; //declaració de l'array de 5 elements String,
els quals encara no s'ha creat (són nuls)
names[0] = new String("John");
names[2] = new String("Martha");
names[2] = new String("Louis");
```

També podem assignar a un element un objecte prèviament construït:

```
String name = "Peter"; //o String name = new String("Peter");
names[1] = name;
```

Arrays i mètodes

Els arrays es poden passar com a paràmetres als mètodes i també poden ser retornats per ells.

Passar array a mètode

```
public void printNames(String [] names) {
    for (int i=0; i<names.length; i++) {
        System.out.println(names[i]);
    }
}

String [] listOfNames = {"John", "Martha", "Louis"};
printNames(listOfNames);
```

Retornar arrays des d'un mètode

```
public int [] getNotes() {
    //obtenir les notes
    return new int[] {3, 8, 5};
}

int [] notes = getNotes();
for (int i=0; i<notes.length; i++) {
    System.out.println(notes[i]);
}
```

Recorregut d'arrays amb bucle for-each

```
int[] intArray = { 12, 32, 95, 11, 10 };
for (int elem: intArray) {
    System.out.println(elem);
}
```

Aquests bucles són molt pràctics però no permeten modificar els elements (només la variable d'iteració) ni donen accés a l'índex de l'element.

Exemples

Càlculs estadístics amb arrays

Programa que entra un array d'enters, el mostra i calcula la suma dels seus elements, el valor mínim i el valor màxim.

Primera versió amb tot el codi a la mateixa classe i al principal.

```
import java.util.Scanner;

/**
 * Entra un array, el mostra i calcula suma, mínim i màxim
 * Versió sense funcions
 * @author Jose
 */
public class Ex01a {

    public static void main(String[] args) {
        Scanner lector = new Scanner(System.in);
        //preguntar quants elements
        System.out.print("Quants elements? ");
        int numElements = lector.nextInt();
        //declarar i instanciar l'array
        int [] llista = new int[numElements];
        //llegir els elements
        for (int i=0; i<numElements; i++) {
            //llegir element
            System.out.print("Element "+i+": ");
            int elem = lector.nextInt();
            //posar-lo a la llista
            llista[i] = elem;
        }
        //mostrar la llista
        for (int i=0; i<numElements; i++) {
            System.out.print(" "+llista[i]);
        }
        System.out.println("");
        //calcular suma
        int suma = 0;
        for (int i=0; i<numElements; i++) {
            suma += llista[i];
        }
        System.out.println("Suma: "+suma);
        //calcular mínim
        int minim = llista[0];
        for (int i=0; i<numElements; i++) {
            if (llista[i] < minim) {
                minim = llista[i];
            }
        }
        System.out.println("Mínim: "+minim);
        //calcular màxim
        int maxim = llista[0];
        for (int i=0; i<numElements; i++) {
            if (maxim < llista[i]) {
                maxim = llista[i];
            }
        }
    }
}
```

```
        }  
    }  
    System.out.println("Màxim: "+maxim);  
}  
}
```

Segona versió amb funcions a la classe principal

```
import java.util.Scanner;  
  
/**  
 * Entra un array, el mostra i calcula suma, mínim i màxim  
 * Versió amb funcions a la mateixa classe  
 * @author Jose  
 */  
public class Ex01b {  
  
    public static void main(String[] args) {  
        Scanner lector = new Scanner(System.in);  
        //preguntar quants elements  
        System.out.print("Quants elements? ");  
        int numElements = lector.nextInt();  
        //declarar i llegir la llista  
        int [] llista = llegirLlista(numElements);  
        //mostrar la llista  
        mostrarLlista(llibra);  
        //calcular suma  
        int suma = calcularSuma(llibra);  
        System.out.println("Suma: "+suma);  
        //calcular mínim  
        int minim = calcularMinim(llibra);  
        System.out.println("Mínim: "+minim);  
        //calcular màxim  
        int maxim = calcularMaxim(llibra);  
        System.out.println("Màxim: "+maxim);  
    }  
  
    /**  
     * llegeix una llista d'enters de la longitud especificada  
     * @param longitud nombre d'elements a llegir  
     * @return array amb la llista  
     */  
    public static int [] llegirLlista(int longitud) {  
        Scanner lector = new Scanner(System.in);  
        //declarar i instanciar l'array  
        int [] dades = new int[longitud];  
        //llegir els elements  
        for (int i=0; i<longitud; i++) {  
            //llegir element  
            System.out.print("Element "+i+": ");  
            int elem = lector.nextInt();  
            //posar-lo a la llista  
            dades[i] = elem;  
        }  
        return dades;  
    }  
}
```

```
}

/**
 * mostra per pantalla una array d'enters
 * @param dades l'array a mostrar
 */
public static void mostrarLlista(int [] dades) {
    //mostrar la llista
    for (int i=0; i<dades.length; i++) {
        System.out.print(" "+dades[i]);
    }
    System.out.println("");
}

/**
 * calcula la suma dels elements de l'array
 * @param dades l'array els elements del qual cal sumar
 * @return la suma dels elements de l'array
 */
public static int calcularSuma(int [] dades) {
    int suma = 0;
    for (int i=0; i<dades.length; i++) {
        suma += dades[i];
    }
    return suma;
}

/**
 * calcula el mínim dels elements de l'array
 * @param dades l'array del qual cal calcular el mínim
 * @return el mínim dels elements de l'array
 */
public static int calcularMinim(int [] dades) {
    int minim = dades[0];
    for (int i=0; i<dades.length; i++) {
        if (dades[i] < minim) {
            minim = dades[i];
        }
    }
    return minim;
}

/**
 * calcula el màxim dels elements de l'array
 * @param dades l'array del qual cal calcular el màxim
 * @return el màxim dels elements de l'array
 */
public static int calcularMaxim(int [] dades) {
    int maxim = dades[0];
    for (int i=0; i<dades.length; i++) {
        if (dades[i] > maxim) {
            maxim = dades[i];
        }
    }
    return maxim;
}
}
```


Tercera versió amb els mètodes a una classe separada que fa de biblioteca de funcions.

```
import java.util.Scanner;

/**
 * Entra un array, el mostra i calcula suma, mínim i màxim
 * Versió amb funcions a una altra classe
 * @author Jose
 */
public class Ex01c {

    public static void main(String[] args) {
        Scanner lector = new Scanner(System.in);
        //preguntar quants elements
        System.out.print("Quants elements? ");
        int numElements = lector.nextInt();
        //declarar i llegir la llista
        int [] llista = llegirLlista(numElements);
        //mostrar la llista
        mostrarLlista(llista);
        //calcular suma
        int suma = Estadistica.calcularSuma(llista);
        System.out.println("Suma: "+suma);
        //calcular mínim
        int minim = Estadistica.calcularMinim(llista);
        System.out.println("Mínim: "+minim);
        //calcular màxim
        int maxim = Estadistica.calcularMaxim(llista);
        System.out.println("Màxim: "+maxim);
    }

    /**
     * llegeix una llista d'enters de la longitud especificada
     * @param longitud nombre d'elements a llegir
     * @return array amb la llista
     */
    public static int [] llegirLlista(int longitud) {
        Scanner lector = new Scanner(System.in);
        //declarar i instanciar l'array
        int [] dades = new int[longitud];
        //llegir els elements
        for (int i=0; i<longitud; i++) {
            //llegir element
            System.out.print("Element "+i+": ");
            int elem = lector.nextInt();
            //posar-lo a la llista
            dades[i] = elem;
        }
        return dades;
    }

    /**
     * mostra per pantalla un array d'enters
     * @param dades l'array a mostrar
     */
    public static void mostrarLlista(int [] dades) {
        //mostrar la llista
    }
}
```

```
        for (int i=0; i<dades.length; i++) {
            System.out.print(" "+dades[i]);
        }
        System.out.println("");
    }
}
```

on la classe biblioteca és aquesta

```
/**
 * Classe per fer càlculs estadístics amb arrays
 *
 * @author Jose
 */
public class Estadistica {

    /**
     * calcula la suma dels elements de l'array
     *
     * @param dades l'array els elements del qual cal sumar
     * @return la suma dels elements de l'array
     */
    public static int calcularSuma(int[] dades) {
        int suma = 0;
        for (int i = 0; i < dades.length; i++) {
            suma += dades[i];
        }
        return suma;
    }

    /**
     * calcula el mínim dels elements de l'array
     *
     * @param dades l'array del qual cal calcular el mínim
     * @return el mínim dels elements de l'array
     */
    public static int calcularMinim(int[] dades) {
        int minim = dades[0];
        for (int i = 0; i < dades.length; i++) {
            if (dades[i] < minim) {
                minim = dades[i];
            }
        }
        return minim;
    }

    /**
     * calcula el màxim dels elements de l'array
     *
     * @param dades l'array del qual cal calcular el màxim
     * @return el màxim dels elements de l'array
     */
    public static int calcularMaxim(int[] dades) {
        int maxim = dades[0];
        for (int i = 0; i < dades.length; i++) {
            if (maxim < dades[i]) {
                maxim = dades[i];
            }
        }
        return maxim;
    }
}
```

```
        }  
    }  
    return maxim;  
}  
  
}
```

Exercici proposat: Afegir a la classe *Estadistica.java* mètodes per calcular la mitjana aritmètica i la desviació estàndard. Provar-ne el funcionament invocant-les des de la classe principal.

La classe Arrays

La classe [java.util.Arrays](#) conté nombrosos mètodes estàtics per manipular arrays.

[Tutorial d'ús de la classe Arrays](#)

A tall d'exemple (T representa qualsevol tipus de dades):

`static String toString(T[] a)` retorna una representació en format *String* de l'array.

`static void sort(T[] a)` ordena en ordre ascendent l'array.

`static void fill(T[] a, T val)` assigna el valor *val* a tots els elements de l'array.

```
int [] dades = {10, 6, 2, 5, 7};  
System.out.println( Arrays.toString(dades) ); //mostra [10, 6, 2, 5, 7]  
Arrays.sort( dades ); //ordena l'array dades de menor a major  
System.out.println( Arrays.toString(dades) ); //mostra [2, 5, 6, 7, 10]  
dades = Arrays.fill( dades, 9 ); //omple l'array dades amb 9's  
System.out.println( Arrays.toString(dades) ); //mostra [9, 9, 9, 9, 9]
```

Taules i estructures multidimensionals

Arrays amb dimensions fixes

Els arrays multidimensionals en Java consisteixen en arrays d'arrays, és a dir, arrays on cada element és a la seva vegada un array.

Ens centrarem en aquest apartat sobretot en arrays bidimensionals, també anomenats taules o matrius.

```
//Two dimensional array (10x20 elements):  
int[][] arr2D = new int[10][20];
```

```
//Three dimensional array (10x15x5 elements):  
int[][][] arr3D = new int[10][15][5];
```

Matriz = {{0,2,4,6},{8,10,12,14},{16,18,20,22}};



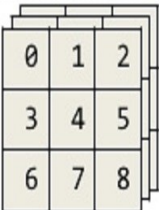
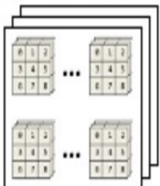
The diagram shows a 3x4 matrix with rows indexed 0 to 2 and columns indexed 0 to 3. The values are as follows:

		Columns			
		0	1	2	3
Filas	0	0	2	4	6
	1	8	10	12	14
	2	16	18	20	22

Matriz [3][4]

El nombre d'elements emmagatzemats en un array multidimensional s'obté fent el producte de totes les seves longituds.

What is an array?

Dimensions	Example	Terminology
1		Vector
2		Matrix
3		3D Array (3 rd order Tensor)
N		ND Array

L'accés als elements es fa amb els índexs de cada dimensió (en ordre). En el cas d'arrays bidimensionals podem imaginar-los com taules o matrius organitzats en files (primer índex) i columnes (segon índex).

```
int[][] arr2D = new int[3][2];
arr2D[0][1] = 3; //element a la primera fila, segona columna.
```

Es poden inicialitzar amb la notació abreujada {}.

```
int[][] arr2D = { {1, 2}, {3, 4}, {5, 6} };
for (int i=0; i<3; i++) { //i recorre les files (primera dimensió)
    for (int j=0; j<2; j++) { //j recorre les columnes (segona dimensió)
        System.out.format("arr[%d][%d]=%d", i, j, arr2D[i][j]);
    }
}
```

O també es poden inicialitzar en qualsevol lloc del codi instanciant un array per a l'element corresponent.

```
int [][] arr3x2 = new int[3][2];
int [] arr1x2 = new int[2];
arr1x2[0] = 1;
```

```
arr1x2[1] = 2;
arr3x2[0][0] = arr1x2;
//el mateix amb les altres files (1 i 2)
```

o també

```
int [][] arr3x2 = new int[3][2];
int [] arr1x2 = {1, 2};
arr3x2[0][0] = arr1x2;
//el mateix amb les altres files (1 i 2)
```

Exemples amb arrays bidimensionals

```
import java.util.Arrays;
import java.util.Random;
```

```
/**
 *
 * @author Jose
 */
public class Matrius {

    public static void main(String[] args) {
        //declarar matriu i generar-la i inicialitzar-la amb un mètode
        int [][] matriu = generarMatriuAleatoria(3, 4);
        //  mostrar matriu en format unilínia
        System.out.println(matrixToString(matriu));
        //declarar i instanciar matriu i inicialitzar-la amb un mètode
        int [][] matriu2 = new int [3][4];
        inicialitzarMatriuAleatoria(matriu2);
        //  mostrar matriu en format unilínia
        System.out.println(matrixToString(matriu2));
        //mostrar matriu en format taula
        System.out.println(matrixToTable(matriu));
    }

    /**
     * declara, instancia i inicialitza amb dades aleatòries
     * una matriu amb les files i columnes especificades
     * @param files el número de files de la matriu
     * @param columnes el número de columnes de la matriu
     * @return la matriu amb les dades
     */
    public static int [][] generarMatriuAleatoria(int files, int columnes) {
        int [][] dades = new int[files][columnes];
        final int MAX_VALOR = 100;
        Random rnd = new Random();
        for (int i = 0; i < dades.length; i++) {
            for (int j = 0; j < dades[i].length; j++) {
                dades[i][j] = rnd.nextInt(MAX_VALOR);
            }
        }
        return dades;
    }

    /**
     * inicialitza una matriu amb dades aleatòries
```

```
* com que el paràmetre 'dades' és una referència,
* es pot usar per accedir als elements de la matriu
* i canviar els seus valors.
* @param dades la matriu a omplir
*/
public static void inicialitzarMatriuAleatoria(int [][] dades) {
    final int MAX_VALOR = 100;
    Random rnd = new Random();
    for (int i = 0; i < dades.length; i++) {
        for (int j = 0; j < dades[i].length; j++) {
            dades[i][j] = rnd.nextInt(MAX_VALOR);
        }
    }
}

/**
 * genera un format String unilínia de la matriu
 * @param dades la matriu a representar
 * @return String amb la matriu en format unilínia
 */
public static String matrixToString(int [][] dades) {
    StringBuilder sb = new StringBuilder();
    sb.append("[");
    for (int i = 0; i < dades.length; i++) {
        sb.append(Arrays.toString(dades[i]));
        if (i < dades.length - 1) sb.append(", ");
    }
    sb.append("]");
    return sb.toString();
}

/**
 * genera un format tabular per a una matriu
 * @param dades la matriu a tabular
 * @return format String tabular de la matriu
 */
public static String matrixToTable(int [][] dades) {
    StringBuilder sb = new StringBuilder();
    //sb.append("[");
    for (int i = 0; i < dades.length; i++) {
        for (int j = 0; j < dades[i].length; j++) {
            sb.append(dades[i][j]);
            if (j < dades[i].length - 1) sb.append("\t");
        }
        if (i < dades.length - 1) sb.append("\n");
    }
    //sb.append("]");
    return sb.toString();
}

}

//llegir array bidimensional de teclat
Scanner lector = new Scanner(System.in);
System.out.print("Nombre de files: ");
int numFiles = lector.nextInt();
System.out.print("Nombre de columnes: ");
int numColumnes = lector.nextInt();
```

```
//declarar i instanciar la matriu de dades
int [][] dades = new int[numFiles][numColumnes];
//llegir dades de l'usuari
for (int i = 0; i < numFiles; i++) { //bucle per recórrer les files
    for (int j = 0; j < numColumnes; j++) { //bucle per recórrer les columnes
        (cel·les de cada fila)
        System.out.format("dades[%d][%d]: ", i, j);
        dades[i][j] = lector.nextInt();
    }
}
```