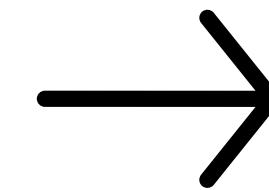


# AEA3

## Programació orientada a objectes



**Prat**

DAM

# AEA3

<b>Nom de l'AEA:</b> Programació orientada a objectes	<b>Hores totals AEA:</b> 78 hores									
<b>Descripció:</b> L'alumnat s'introduirà als principis de la programació orientada a objectes (POO)										
<b>Resultat d'Aprendentatge</b>										
RA2: Escriu i prova programes senzills, reconeixent i aplicant els fonaments de la programació orientada a objectes. RA4: Desenvolupa programes organitzats en classes analitzant i aplicant els principis de la programació orientada a objectes. RA5: Realitza operacions d'entrada i sortida d'informació, utilitzant procediments específics del llenguatge i llibreries de classes. RA7: Desenvolupa programes aplicant característiques avançades dels llenguatges orientats a objectes i de l'entorn de programació.										
<b>Organitzadors previs:</b> Per desenvolupar aquesta AEA necessitem haver assolits els criteris d'avaluació de les AEA anteriors										
<b>Seqüència de l'activitat d'ensenyament-aprenentatge</b>	<b>Organització aula</b>	<b>Temporització</b>	<b>Metodologia</b>	<b>Instruments d'avaluació</b>	<b>Localització</b>					
<b>AA1 - Imprimir chars</b> 5%	Individual	3	Aprenentatge cooperatiu	Rúbrica	Aula					
<b>AA2 - Llegir reals</b> 5%	Individual	5	Aprenentatge cooperatiu	Rúbrica	Aula					
<b>AA3 - Càcul àrees amb funcions</b> 10%	Individual	5	ABP	Rúbrica	Aula					
<b>AA4 - Càcul nota màxima, mínima i mitjana amb funcions</b> 10%	Individual	10	ABP	Rúbrica	Aula					

# AEA3

<b>Nom de l'AEA:</b> Programació orientada a objectes			<b>Hores totals AEA:</b> 78 hores		
<b>Seqüència de l'activitat d'ensenyament-aprenentatge</b>	<b>Organització aula</b>	<b>Temporització</b>	<b>Metodologia</b>	<b>Instruments d'avaluació</b>	<b>Localització</b>
<b>AA5 - Empresa 5%</b>	Individual	5	Aprenentatge cooperatiu	Rúbrica	Aula
<b>AA6 - Persona 5%</b>	Parelles	10	Aprenentatge cooperatiu	Rúbrica	Aula
<b>PR1 - Figura 30%</b>	Individual	20	ABP	Rúbrica	Aula
<b>PR2 - Sistema de reserves 30%</b>	Individual	20	ABP	Rúbrica	Aula

# Mètodes o funcions

# Mètodes

Fins ara, tots els programes que hem creat els hem desenvolpat dins la funció main.

Un canvi necessari en el format dels vostres programes quan es vol declarar altres mètodes, és a dir, el que farem a partir d'ara és **dividir el programa**.

# Mètodes

## Exemple

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
public void inici() {  
    llegirLlista();  
    ordenarLlista();  
    mostrarLlista();  
}  
//Mètode amb les instruccions per llegir la llista.  
//El primer valor sera la llargària  
public void llegirLlista() {  
    System.out.println("Escriu una llista de valors enters i prem retorn.");  
    System.out.println("El primer valor indica la mida de la seqüència.");  
    llegirMida();  
    llegirValors();  
}  
public void llegirMida() {//Mètode que llegeix el primer valor  
    //Lectura  
    int mida = 0;  
    if (lector.hasNextInt()) {  
        mida = lector.nextInt();  
    } else {  
        lector.next();  
    }  
    llistaEnters = new int[mida]; //Inicialització diferida de l'array  
}  
public void llegirValors() {  
    int index = 0;  
    while (index < llistaEnters.length) {  
        if (lector.hasNextInt()) {  
            llistaEnters[index] = lector.nextInt();  
            index++;  
        } else {  
            lector.next();  
        }  
    }  
    lector.nextLine();  
} //La resta de mètodes no canvien ...
```

} Funció Main —> Mètode que hem utilitzat fins ara

} Funció Inici —> Primer mètode

} Funció llegirLlista —> Segon mètode

} Funció llegirMida —> Tercer mètode

} Funció llegirValors —> Quart mètode

# Variable global

Una variable global és una variable que pot ser accedida des de qualsevol instrucció dins un mateix fitxer de codi font. **El seu àmbit és tot el fitxer.**

> Sintaxi:

```
private tipus nomVariable = valorInicial;
```

Exemple

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
  
public void inici() {  
    llegirLlista();  
    ordenarLlista();  
    mostrarLlista();  
}  
  
//Mètode amb les instruccions per llegir la llista.  
//El primer valor sera la llargària  
public void llegirLlista() {  
    System.out.println("Escriu una llista de valors enters i prem retorn.");  
    System.out.println("El primer valor indica la mida de la seqüència.");  
    llegirMida();  
    llegirValors();  
}  
public void llegirMida() {  
    //Metode que llegeix el primer valor  
}
```

# Invocació d'un mètode

Invoke un mètode vol dir fer que el programa l'executi.

> Sintaxi:

nomMetode( );

Exemple

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
public void inici() {  
    lleqirLlista();  
    ordenarLlista();  
    mostrarLlista();  
}  
//Mètode amb les instruccions per llegir la llista.  
//El primer valor sera la llargària  
public void lleqirLlista() {  
    System.out.println("Escriu una llista de valors enters i prem retorn.");  
    System.out.println("El primer valor indica la mida de la seqüència.");  
    llegirMida();  
    llegirValors();  
}  
public void llegirMida() {  
    //Metode que llegeix el primer valor
```

# Mètodes

Exemple

Què fa el  
programa?  
Quin ordre  
segueix?

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
1  public void inici() {  
    llegirLlista();  
    ...  
}  
//Mètode amb les instruccions per llegir la llista.  
//El primer valor sera la llargària  
public void llegirLlista() {  
    System.out.println("Escriu una llista de valors enters i prem retorn.");  
    System.out.println("El primer valor indica la mida de la seqüència.");  
    llegirMida();  
    llegirValors();  
}  
public void llegirMida() {           //Metode que llegeix el primer valor  
    //Lectura  
    int mida = 0;  
    if (lector.hasNextInt()) {  
        mida = lector.nextInt();  
    } else {  
        lector.next();  
    }  
    llistaEnters = new int[mida];   //Inicialització diferida de l'array  
}  
public void llegirValors() {  
    int index = 0;  
    while (index < llistaEnters.length) {  
        if (lector.hasNextInt()) {  
            llistaEnters[index] = lector.nextInt();  
            index++;  
        } else {  
            lector.next();  
        }  
    }  
    lector.nextLine();  
}                                //La resta de mètodes no canvien ...  
}
```

# Mètodes

Exemple

Què fa el  
programa?  
Quin ordre  
segueix?

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
1  public void inici() {  
    llegirLlista();  
      
      
2  }  
    //Mètode amb les instruccions per llegir la llista.  
    //El primer valor sera la llargària  
    public void llegirLlista() {  
        System.out.println("Escriu una llista de valors enters i prem retorn.");  
        System.out.println("El primer valor indica la mida de la seqüència.");  
        llegirMida();  
        llegirValors();  
    }  
    public void llegirMida() {          //Metode que llegeix el primer valor  
        //Lectura  
        int mida = 0;  
        if (lector.hasNextInt()) {  
            mida = lector.nextInt();  
        } else {  
            lector.next();  
        }  
        llistaEnters = new int[mida];  //Inicialització diferida de l'array  
    }  
    public void llegirValors() {  
        int index = 0;  
        while (index < llistaEnters.length) {  
            if (lector.hasNextInt()) {  
                llistaEnters[index] = lector.nextInt();  
                index++;  
            } else {  
                lector.next();  
            }  
            lector.nextLine();  
        }  
    }  
    //La resta de mètodes no canvien ...  
}
```

# Mètodes

Exemple

Què fa el  
programa?  
Quin ordre  
segueix?

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
1  public void inici() {  
    llegirLlista();  
}  
2  }  
   //Mètode amb les instruccions per llegir la llista.  
   //El primer valor sera la llargària  
   public void llegirLlista() {  
        System.out.println("Escriu una llista de valors enters i prem retorn.");  
        System.out.println("El primer valor indica la mida de la seqüència.");  
        llegirMida();  
        llegirValors();  
    }  
3  }  
   public void llegirMida() {      //Metode que llegeix el primer valor  
        //Lectura  
        int mida = 0;  
        if (lector.hasNextInt()) {  
            mida = lector.nextInt();  
        } else {  
            lector.next();  
        }  
        llistaEnters = new int[mida];  //Inicialització diferida de l'array  
    }  
   public void llegirValors() {  
        int index = 0;  
        while (index < llistaEnters.length) {  
            if (lector.hasNextInt()) {  
                llistaEnters[index] = lector.nextInt();  
                index++;  
            } else {  
                lector.next();  
            }  
            lector.nextLine();  
        }  
    }  
}                                //La resta de mètodes no canvien ...
```

# Mètodes

Exemple

Què fa el  
programa?  
Quin ordre  
segueix?

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
1  public void inici() {  
    llegirLlista();  
}  
2  }  
   //Mètode amb les instruccions per llegir la llista.  
   //El primer valor sera la llargària  
   public void llegirLlista() {  
        System.out.println("Escriu una llista de valors enters i prem retorn.");  
        System.out.println("El primer valor indica la mida de la seqüència.");  
        llegirMida();  
        llegirValors();  
    }  
3  }  
   public void llegirMida() {  
        //Lectura  
        int mida = 0;  
        if (lector.hasNextInt()) {  
            mida = lector.nextInt();  
        } else {  
            lector.next();  
        }  
        llistaEnters = new int[mida]; //Inicialització diferida de l'array  
    }  
4  public void llegirValors() {  
        int index = 0;  
        while (index < llistaEnters.length) {  
            if (lector.hasNextInt()) {  
                llistaEnters[index] = lector.nextInt();  
                index++;  
            } else {  
                lector.next();  
            }  
            lector.nextLine();  
        }  
    }  
} //La resta de mètodes no canvien ...
```

Un cop executa tot el codi, tornem a la funció que l'ha cridat.

# Mètodes

Exemple

Què fa el  
programa?  
Quin ordre  
segueix?

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
1 public void inici() {  
    llegirLlista();  
}  
2 }  
//Mètode amb les instruccions per llegir la llista.  
//El primer valor sera la llargària  
public void llegirLlista() {  
    System.out.println("Escriu una llista de valors enters i prem retorn.");  
    System.out.println("El primer valor indica la mida de la seqüència.");  
    llegirMida();  
    llegirValors();  
}  
3 }  
public void llegirMida() {  
    //Lectura  
    int mida = 0;  
    if (lector.hasNextInt()) {  
        mida = lector.nextInt();  
    } else {  
        lector.next();  
    }  
    llistaEnters = new int[mida]; //Inicialització diferida de l'array  
}  
4 //Metode que llegeix el primer valor  
5 public void llegirValors() {  
    int index = 0;  
    while (index < llistaEnters.length) {  
        if (lector.hasNextInt()) {  
            llistaEnters[index] = lector.nextInt();  
            index++;  
        } else {  
            lector.next();  
        }  
        lector.nextLine();  
    }  
} //La resta de mètodes no canvien ...  
}
```

# Mètodes

Exemple

Què fa el  
programa?  
Quin ordre  
segueix?

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
1 public void inici() {  
    llegirLlista();  
}  
2 }  
//Mètode amb les instruccions per llegir la llista.  
//El primer valor sera la llargària  
public void llegirLlista() {  
    System.out.println("Escriu una llista de valors enters i prem retorn.");  
    System.out.println("El primer valor indica la mida de la seqüència.");  
    llegirMida();  
    llegirValors();  
}  
3 }  
public void llegirMida() {  
    //Lectura  
    int mida = 0;  
    if (lector.hasNextInt()) {  
        mida = lector.nextInt();  
    } else {  
        lector.next();  
    }  
    llistaEnters = new int[mida]; //Inicialització diferida de l'array  
}  
4 //Metode que llegeix el primer valor  
5 public void llegirValors() {  
    int index = 0;  
    while (index < llistaEnters.length) {  
        if (lector.hasNextInt()) {  
            llistaEnters[index] = lector.nextInt();  
            index++;  
        } else {  
            lector.next();  
        }  
        lector.nextLine();  
    }  
}  
6 //La resta de mètodes no canvien ...  
}
```

Un cop executa tot el codi, tornem a la funció que l'ha cridat.

# Mètodes

Exemple

Què fa el  
programa?  
Quin ordre  
segueix?

```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
public void inici() {  
    llegirLlista();  
}  
}  
//Mètode amb les instruccions per  
//El primer valor sera la llargària  
public void llegirLlista() {  
    System.out.println("Escriu una llista de valors enters i prem retorn.");  
    System.out.println("El primer valor indica la mida de la seqüència.");  
    llegirMida();  
    llegirValors();  
}  
public void llegirMida() {  
    //Lectura  
    int mida = 0;  
    if (lector.hasNextInt()) {  
        mida = lector.nextInt();  
    } else {  
        lector.next();  
    }  
    llistaEnters = new int[mida]; //Inicialització diferida de l'array  
}  
public void llegirValors() {  
    int index = 0;  
    while (index < llistaEnters.length) {  
        if (lector.hasNextInt()) {  
            llistaEnters[index] = lector.nextInt();  
            index++;  
        } else {  
            lector.next();  
        }  
        lector.nextLine();  
    }  
}
```

1      2      3      4      5      6      7

Un cop executa tot el codi, tornem a la funció que l'ha cridat.

    //Metode que llegeix el primer valor

    //La resta de mètodes no canvia ...

# Mètodes

Exemple

Què fa el  
programa?  
Quin ordre  
segueix?

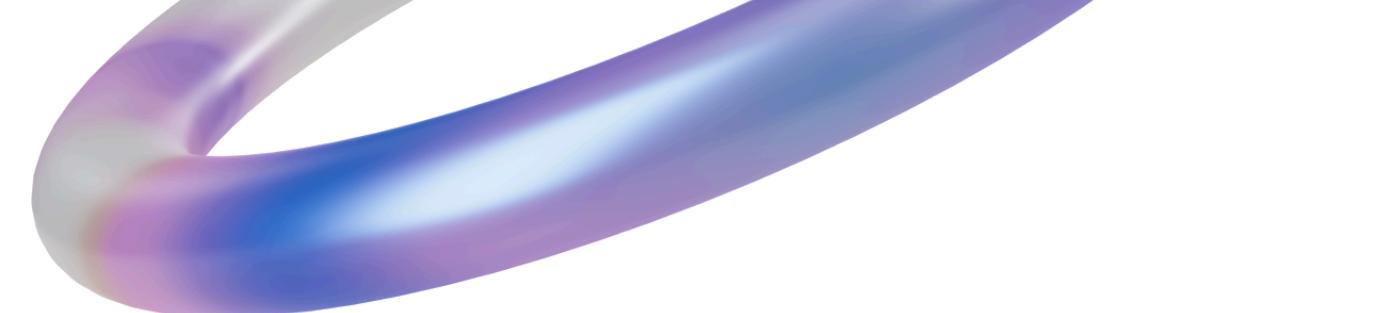
```
//Variable global. Array no inicialitzat.  
private int[] llistaEnters = null;  
//En aplicar disseny descendent, ara cal declarar "lector" com a global  
Scanner lector = new Scanner(System.in);  
public static void main (String[] args) {  
    OrdenarDescendentVariable programa = new OrdenarDescendentVariable();  
    programa.inici();  
}  
1 public void inici() {  
    llegirLlista();  
}  
2 }  
//Mètode amb les instruccions per llegir la llista.  
//El primer valor sera la llargària  
public void llegirLlista() {  
    System.out.println("Escriu una llista de valors enters i prem retorn.");  
    System.out.println("El primer valor indica la mida de la seqüència.");  
    llegirMida();  
3    llegirValors();  
}  
public void llegirMida() {  
    //Lectura  
    int mida = 0;  
    if (lector.hasNextInt()) {  
        mida = lector.nextInt();  
    } else {  
        lector.next();  
    }  
5    llistaEnters = new int[mida]; //Inicialització diferida de l'array  
}  
public void llegirValors() {  
    int index = 0;  
    while (index < llistaEnters.length) {  
        if (lector.hasNextInt()) {  
            llistaEnters[index] = lector.nextInt();  
            index++;  
        } else {  
            lector.next();  
        }  
    }  
    lector.nextLine();  
}  
6 }  
//La resta de mètodes no canvien ...  
7 }  
8 }
```

Un cop executa tot el codi, tornem a la funció que l'ha cridat.

# Programació modular

## Descomposició de problemes

# Programació modular



Exemple

1

**Volem fer un programa que gestioni un registre de temperatures al llarg de l'any.**

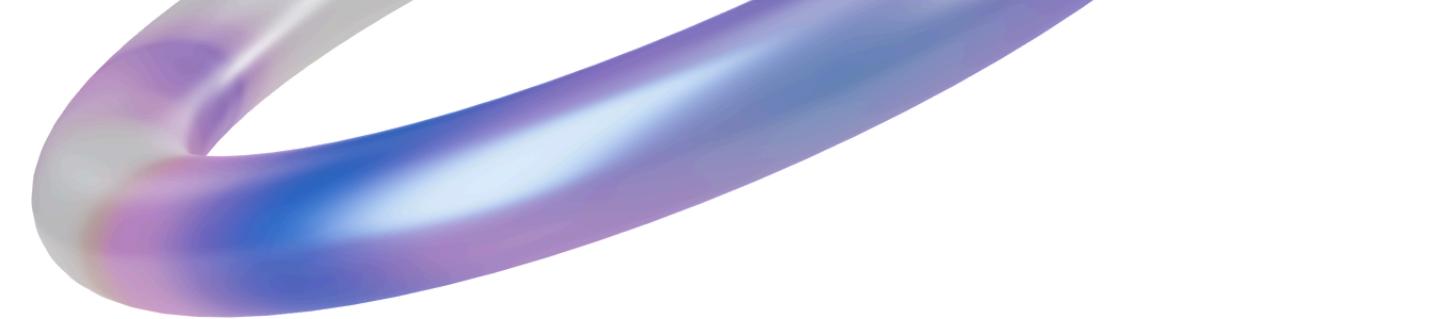
1. Definició de dades

- Les temperatures es guardarán en un array de reals (double[]).
- Cada posició de l'array correspon a un dia de l'any.
- Com que volem treballar amb 52 setmanes, cal que l'array tingui espai per a 364 dies ( $52 * 7$ ).
- També necessitarem dos enters per representar la data:
  - dia (1–31)
  - mes (1–12)

⚠ Cal tenir en compte que no tots els dies tindran una temperatura assignada.

Per exemple, després de la primera setmana només les posicions 0 a 6 contindran dades vàlides.

# Programació modular



2

Exemple

**En aquest programa modular, organitzarem el codi en funcions (mètodes) per fer-lo més clar i mantenible.**

Les dues funcions principals són:

## 1. Mostrar el menú d'opcions

- Aquesta funció mostrarà a l'usuari les diferents accions que pot fer amb el registre de temperatures.
- Per exemple:
  - Afegir una temperatura
  - Consultar la temperatura d'un dia
  - Mostrar totes les temperatures registrades fins ara
  - Sortir del programa
- La funció no rep ni retorna cap valor, només mostra informació per pantalla.

# Programació modular

2

Exemple

## 2. Llegir el que l'usuari demana (tractar ordre)

- Aquesta funció s'encarrega de llegir la tria de l'usuari i executar l'acció corresponent.
- Aquí entra la modularització: cada acció (afegir, consultar, mostrar) es pot implementar en una funció separada.
- Funcions:
  - Registrar temperatures
  - Mostrar temperatura mitjana
  - Mostrar diferència màxima
  - Finalitzar execució



La idea de modularitzar és separar el “què es fa” del “com es fa”, així:

- `mostrarMenu()` → només mostra informació
- `tractarOrdre(opcio, arrayTemperatures)` → només gestiona la selecció de l'usuari
- Funcions internes com `afegirTemperatura()` o `consultarTemperatura()` fan la feina concreta, sense barrejar-se amb el menú.

# Programació modular



3

Exemple

## Dividir el programa en parts més concretes

### 1. Registrar temperatures setmanals

- **Llegir temperatures del teclat.** Aquesta funció llegeix les temperatures d'una setmana (7 dies) des del teclat.
- **Actualitzar la data actual.** Després d'afegir les temperatures, actualitza les variables de dia i setmana per saber quin és l'últim dia registrat.

### 2. Mostrar temperatura mitjana

- **Mostrar data actual.** Mostra quin és l'últim dia registrat o la setmana actual.
- **Calcular temperatura mitjana.** Calcula la mitjana de totes les temperatures registrades fins ara i la mostra a l'usuari.

### 3. Mostrar diferència màxima

- **Mostrar data actual.** Mostra la data o setmana en què es farà el càlcul.
- **Calcular diferència màxima.** Troba la diferència entre la temperatura mínima i la màxima registrada fins al moment i la mostra.

### 4. Finalitzar execució.

Funció que mostra un missatge de comiat i finalitza el programa de manera ordenada.

# Registre temperatures - Part 1

4

## Registre de temperatures

>

1. Descarregeu-vos el codi font anomenat: RegistreTemperatures-Part1.java

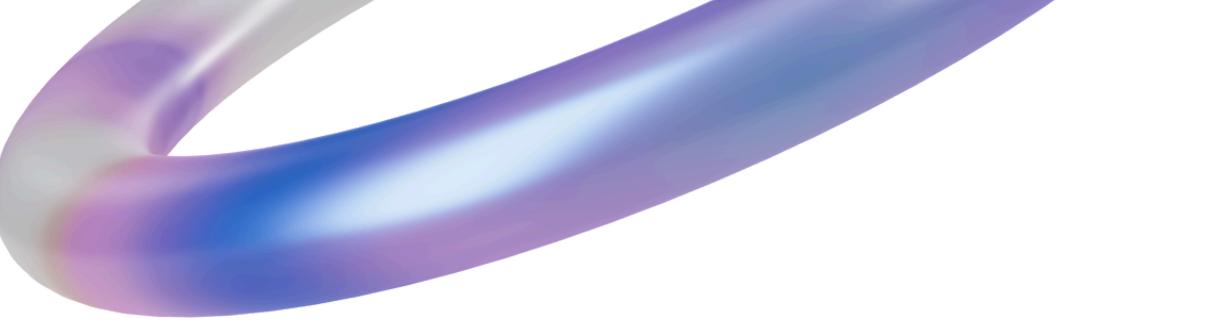
2. **Declareu els següents mètodes dins el codi (no els desenvolpeu**

**encara!).** Els seus noms seran:

- `IlegirTemperaturesTeclat()`
- `incrementarData()`
- `mostrarData()`
- `calculaMitjana()`
- `calculaDiferencia()`

 No cal fer entrega a ALEXIA

# Registre temperatures - Part 2



5

## Registre de temperatures

>

1. Descarregeu-vos el codi font anomenat: RegistreTemperatures-Part2.java

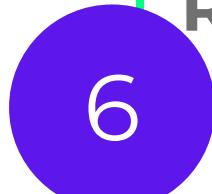
### 2. Analitzeu els mètodes desenvolupats al codi:

- llevarTemperaturesTeclat()
- incrementarData()
- mostrarData()
- calculaMitjana()

3. Desenvolupueu el mètode: **calcularDiferenciaMaxima()**

 No cal fer entrega a ALEXIA

# Registre temperatures - Part 3



6

## Registre de temperatures

>

1. Descarregeu-vos el codi font anomenat: RegistreTemperatures-Part3.java

2. **Desenvolupeu els següents mètodes:**

- registreTemperaturesSetmanals() → la fem tots junts
- mostrarMitjana() → la fem tots junts
- mostrarDiferencia() → la desenvolapeu de forma individual

3. NO desenvolapeu el mètode: **finalitzarExecucio()**

 No cal fer entrega a ALEXIA

# Registre temperatures - Part 4

7

## Registre de temperatures

>

1. Descarregeu-vos el codi font anomenat: RegistreTemperatures-Part4.java

### 2. Desenvolupeu els següents mètodes:

- mostrarMenu() → la desenvolupreu de forma individual
- tractarOpcio() → la desenvolupreu de forma individual

### 3. NO desenvolupeu el mètode: **tractarOpcio()**

⚠️ No cal fer entrega a ALEXIA

The screenshot shows a terminal window with the following text:

funció: mostrarMenu()

Benvingut al registre de temperatures

[RT] Registrar temperatures setmanals.  
[MJ] Consultar temperatura mitjana.  
[DF] Consultar diferència màxima.  
[FI] Sortir.

Opció: MJ

No hi ha temperatures registrades.

opció escrita per l'usuari

resposta del programa

Benvingut al registre de temperatures

[RT] Registrar temperatures setmanals.  
[MJ] Consultar temperatura mitjana.  
[DF] Consultar diferència màxima.  
[FI] Sortir.

Opció: RT

Escriu les temperatures d'aquesta setmana:  
20,5 21,1 21 21,7 20,9 20,6 19,9

Benvingut al registre de temperatures

# Registre temperatures - Part 5

8

## Registre de temperatures

>

1. Descarregeu-vos el codi font anomenat: RegistreTemperatures-Part5.java

### 2. Desenvolupeu els següents mètodes:

- inici() → la desenvolupreu de forma individual
- finalitzarExecucio() → la desenvolupreu de forma individual

 No cal fer entrega a ALEXIA

Benvingut al registre de temperatures

[RT] Registrar temperatures setmanals.

[MJ] Consultar temperatura mitjana.

[DF] Consultar diferència màxima.

[FI] Sortir.

Opció: MJ

No hi ha temperatures registrades.

#### funció: inici()

repeteix el menú fins que l'usuari li indica que s'acaba el programa escrivint "fi"

Benvingut al registre de temperatures

[RT] Registrar temperatures setmanals.

[MJ] Consultar temperatura mitjana.

[DF] Consultar diferència màxima.

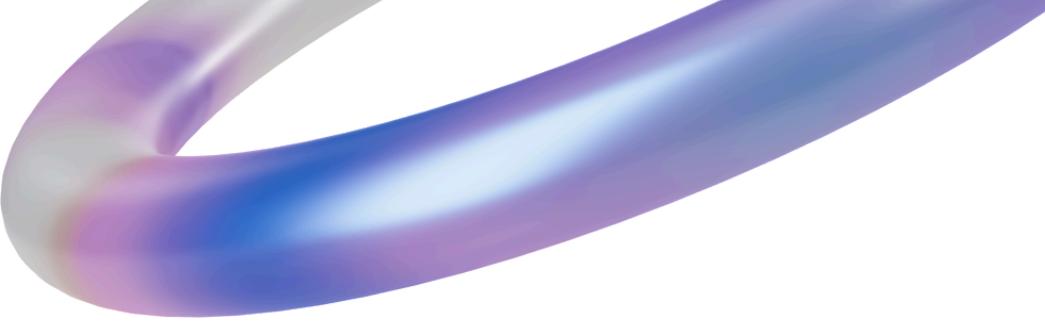
[FI] Sortir.

Opció: RT

Escriviu les temperatures d'aquesta setmana:

20,5 21,1 21 21,7 20,9 20,6 19,9

# Registre temperatures - Part 6



9

## Registre de temperatures

>

1. Descarregeu-vos el codi font anomenat: RegistreTemperatures-Part6.java
  - **Reviseu si podeu millorar la funció/ comportament de la funció finalitzarExecucio()**
2. Entregar el codi final

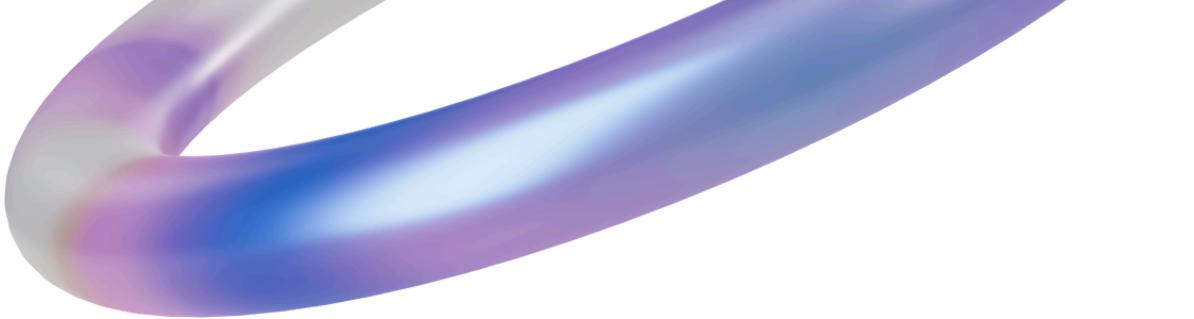
# Programació modular

## Parametrització dels mètodes

# Paràmetres de mètodes

Un **paràmetre** és un **identificador** utilitzat dins la descripció d'un mètode, pot ser un **paràmetre d'entrada o de sortida.**

# Paràmetres de mètodes



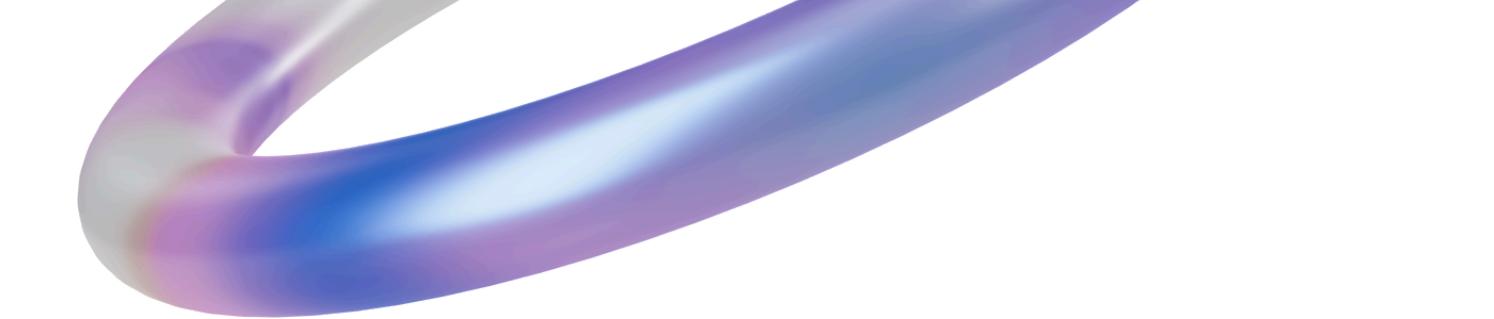
## Paràmetres d'entrada

**Valor** que **s'estableix** immediatament **abans d'executar un procés**, de manera que **indica les dades que ha de tractar o que modifica el seu comportament.**

## Paràmetres de sortida

**Valor** que **indica un resultat final** obtingut **després** de **realitzar** un **procés** determinat.

# Paràmetres de mètodes



Anem a suposar que volem fer una publicació a Instagram i una altra a TikTok.

		Fer publicació a Instagram	
1	Obrir l'app de xarxes socials	-	-
2	Seleccionar el compte	Instagram	-
3	Crear	Publicació a Instagram	-
4	Afegir la descripció	A la publicació a Instagram	-
5	Penjar	La publicació a Instagram	Publicació penjada
6	Tancar al app	-	-

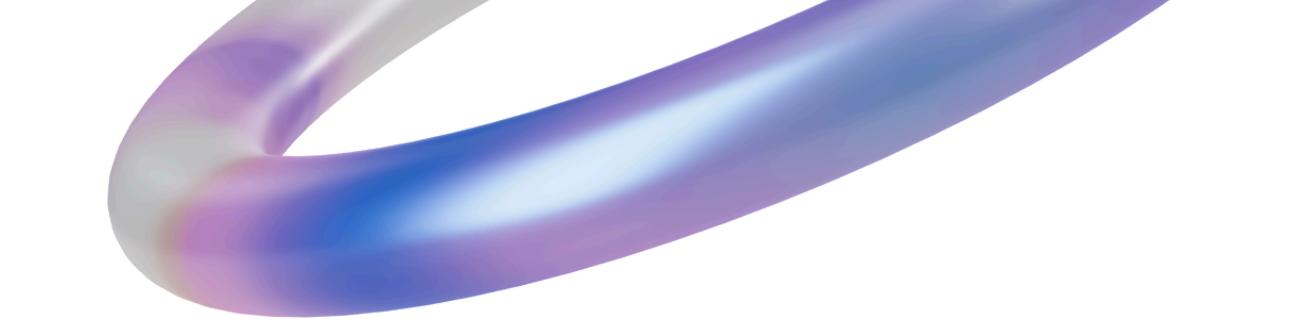
# Paràmetres de mètodes



Anem a suposar que volem fer una publicació a Instagram i una altra a TikTok.

		Fer publicació a TikTok	
1	Obrir l'app de xarxes socials	-	-
2	Seleccionar el compte	TikTok	-
3	Crear	Video a TikTok	-
4	Afegir la descripció	Al video de TikTok	-
5	Penjar	El video de TikTok	Video penjat
6	Tancar al app	-	-

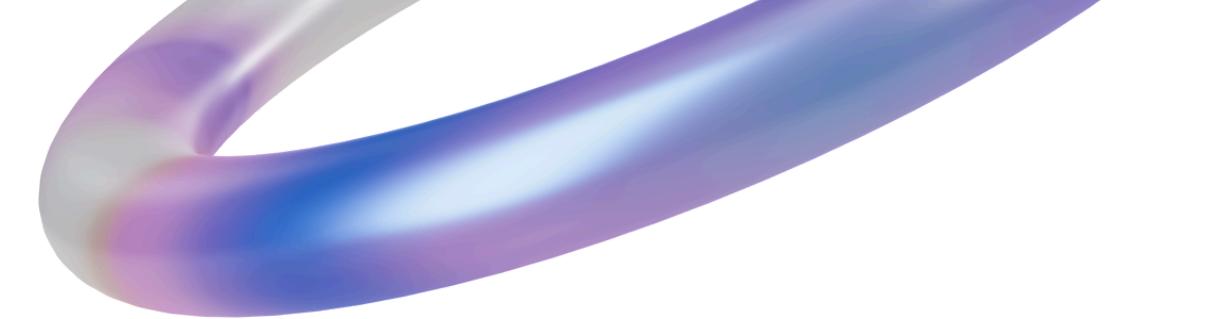
# Paràmetres de mètodes



Anem a suposar que volem fer una publicació a Instagram i una altra a TikTok.

		Fer publicació a Instagram		Fer publicació a TikTok	
		Param entrada	Param sortida	Param entrada	Param sortida
1	Obrir l'app de xarxes socials	-	-	-	-
2	Seleccionar el compte	Instagram	-	TikTok	-
3	Crear	Publicació a Instagram	-	Video a TikTok	-
4	Afegir la descripció	A la publicació a Instagram	-	Al video de TikTok	-
5	Penjar	La publicació a Instagram	Publicació penjada	El video de TikTok	Video penjat
6	Tancar al app	-	-	-	-

# Paràmetres de mètodes



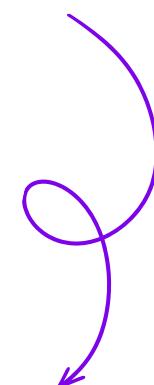
Anem a suposar que volem fer una publicació a instagram i una altra a tiktok.

## Fer publicació a instagram

1. Obrir l'app de xarxes socials
2. Seleccionar el compte **d'Instagram**
3. Crear una **publicació a Instagram**
4. Afegir la descripció i etiquetes a la **publicació d'Instagram**
5. Penjar la **publicació a Instagram**
6. Tancar l'app

## Fer publicació a TikTok

1. Obrir l'app de xarxes socials
2. Seleccionar el compte de **TikTok**
3. Crear un **vídeo a TikTok**
4. Afegir la descripció i efectes al **vídeo de TikTok**
5. Penjar el **vídeo a TikTok**
6. Tancar l'app



- Paràmetre d'entrada: la xarxa social (Instagram, TikTok).
- Procés: els passos generals són els mateixos.
- Paràmetre sortida: la publicació pujada a la xarxa corresponent.

# Paràmetres d'entrada

Declaració de funcions amb paràmetres d'entrada

```
public void nomMètode (tipusParam1 numParam1, tipusParam2 nomParam2, etc.) {  
    //Codi  
    ...  
}
```

Exemple

```
public void imprimirChar(int num, char c){  
    //codi  
}
```

# Paràmetres d'entrada

Innovació de funcions amb paràmetres d'entrada

---

nomMetode(valor1, valor2, etc.);

---

Exemple

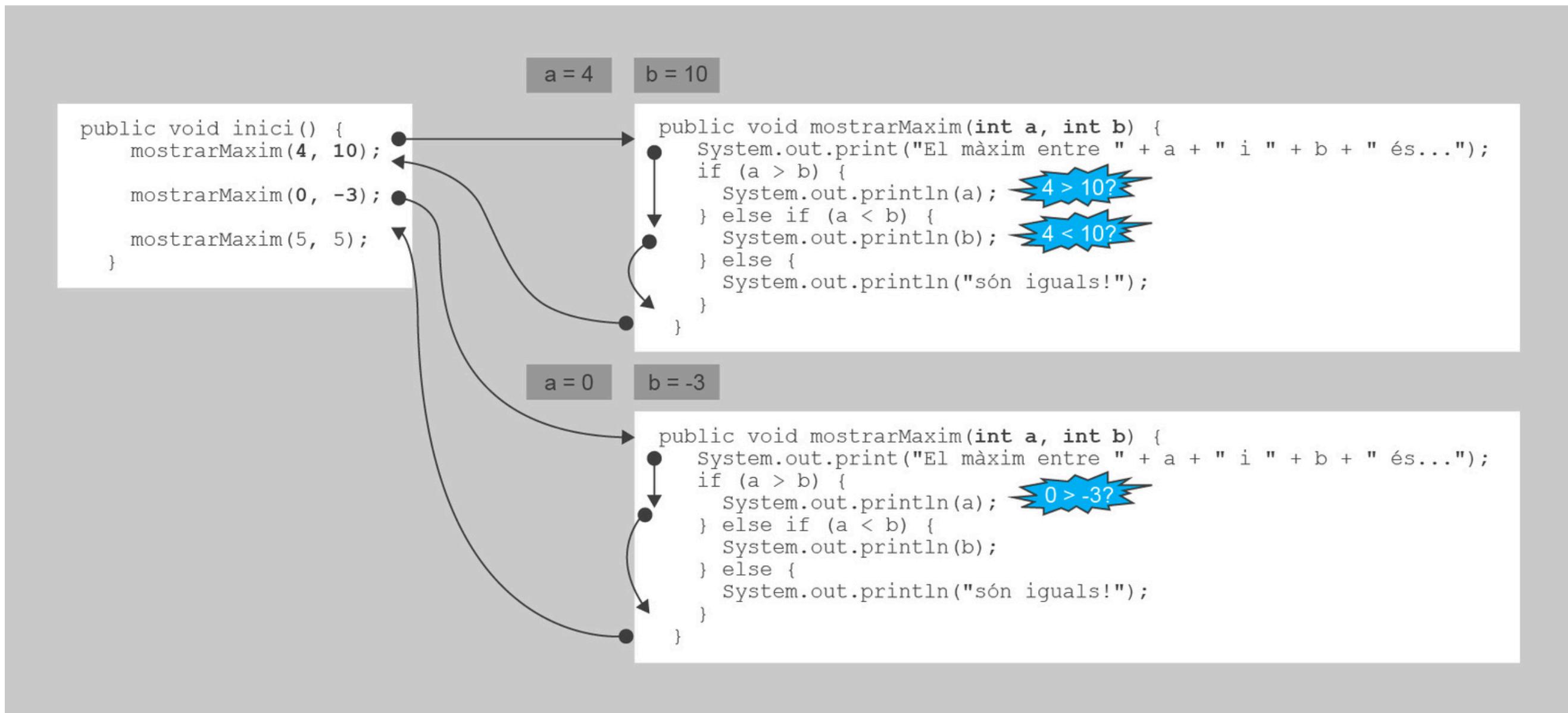
```
public void imprimirChar(int num, char c){  
    //codi  
}
```

```
imprimirChar( num: 6, c: '!' );  
imprimirChar( num: 1, c: 'ξ' );
```

# Paràmetres d'entrada

Exemple

Copieu el codi, compileu-lo i executeu-lo.



# Paràmetres d'entrada



Exercici de classe - Estrelles

**Escriviu un programa que invoqui diverses vegades un mètode que rep un únic paràmetre de tipus enter.** Aquest mètode ha de mostrar per pantalla una línia formada per un nombre de símbols "\*" igual al valor del paràmetre rebut.

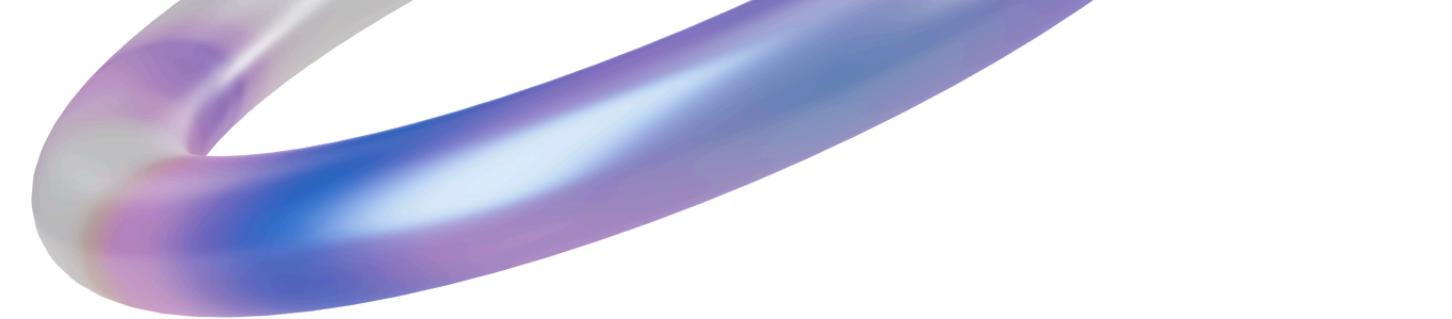
Recordeu que:

- El paràmetre d'entrada és el nombre d'estrelles que vols dibuixar
- Cal implementar una funció que imprimeixi tantes estrelles com s'indiqui per paràmetre d'entrada
- Cal desenvolupar el codi de la funció que dibuixarà les estrelles

La sortida de la terminal ha de ser com aquesta:

```
Has demandat dibuixar...
***funció que imprimeix estrelles
*****
Aquí acaba el programa. Ja he imprés el que m'has demanat
```

# Exercici per entregar



## Imprimir chars

- > Escriviu un programa que cridi diverses vegades un mètode amb dos paràmetres d'entrada:
- El primer paràmetre és un valor de tipus char, que indica quin caràcter s'ha d'imprimir.
  - El segon paràmetre és un valor de tipus int, que indica el nombre de vegades que s'ha d'imprimir aquest caràcter.

El mètode ha de mostrar per pantalla una línia formada pel caràcter indicat, repetit tants cops com indiqui el segon paràmetre.

```
imprimir('*', 5) → *****
imprimir('#', 3) → ####
imprimir('A', 4) → AAAA
```

# Paràmetres de sortida

Declaració de funcions amb paràmetres de sortida

```
public tipusParamSortida nomMètode (llistaParamEntrada) {  
    //Codi  
    ...  
}
```

Exemple

```
Paràmetre de sortida          Paràmetre d'entrada  
public int calcularMaxim (int primerNum, int segonNum){  
    int max = primerNum;  
    if (primerNum < segonNum)  
        max = segonNum;  
    return max;  Cal afegir la instrucció return per a que la funció  
}                                retorni, al mètode original, un valor.
```

# Paràmetres de sortida

Declaració de funcions amb paràmetres de sortida

```
public tipusParamSortida nomMètode (llistaParamEntrada) {  
    //Codi  
    ...  
}
```

Exemple

```
Paràmetre de sortida  
public int calcularMaxim (int primerNum, int segonNum){  
    int max = primerNum;  
    if (primerNum < segonNum)  
        max = segonNum;  
    return max; }  
Paràmetre d'entrada  
Cal afegir la instrucció return per a que la funció  
retorni, al mètode original, un valor.
```

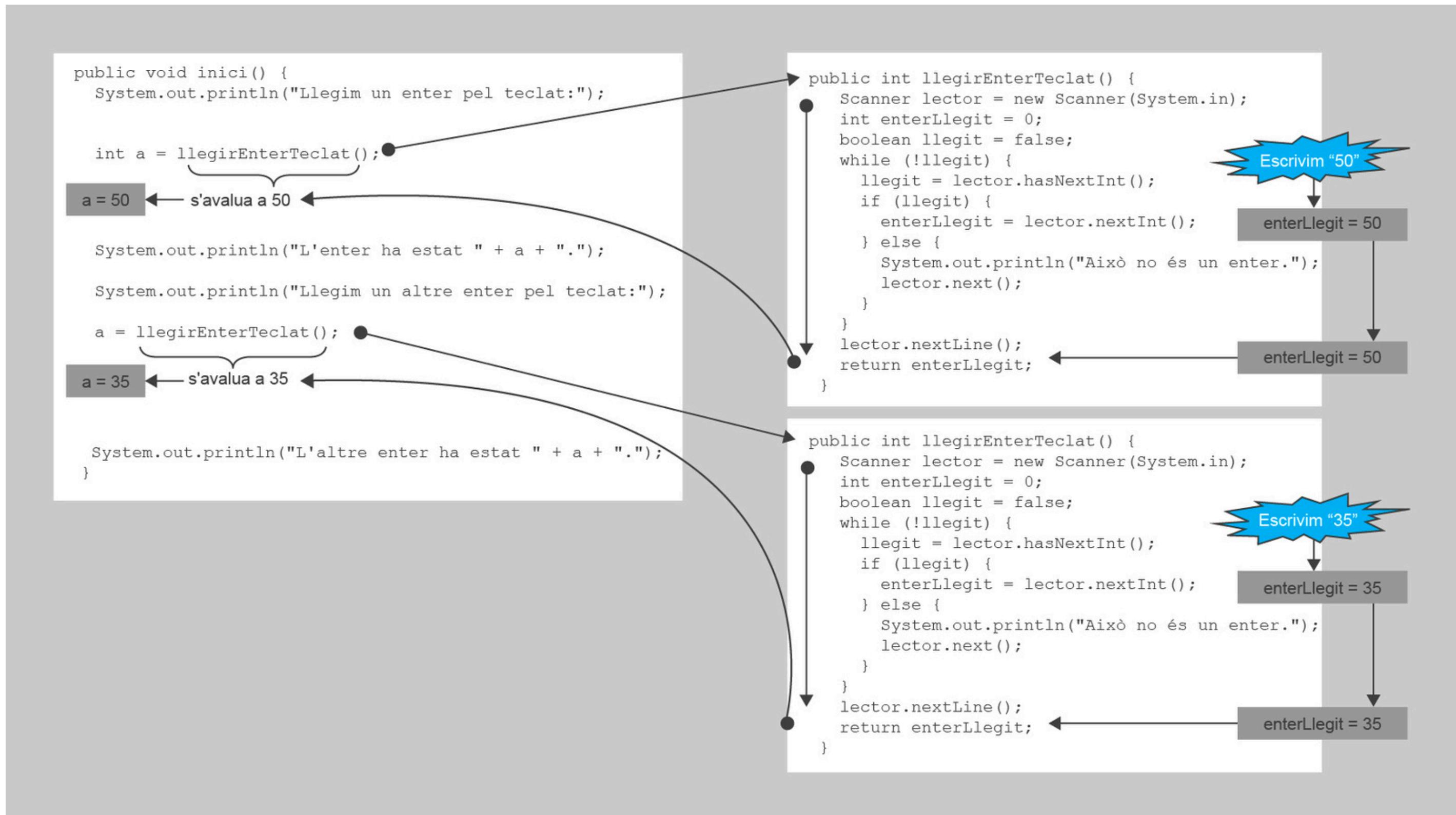
Funció main

```
int max = calcularMaxim( primerNum: 3, segonNum: 4);  
System.out.println(max);
```

# Paràmetres de sortida

Exemple

Copieu el codi,  
compileu-lo i  
executeu-lo.



# Paràmetres de sortida

Exemple

Revisem la funció `llegirEnterTeclat()`:

- **Paràmetre d'entrada?**
- **Paràmetre de sortida?**
  - Té **return**?
- **Tipus del paràmetre de sortida?**
- **Què fa la funció?**

```
public static void main (String[] args) {  
    LlegirEnters programa = new LlegirEnters();  
    programa.inici();  
}  
public void inici() {  
    System.out.println("Llegiu un enter pel teclat:");  
    int a = llegirEnterTeclat();  
    System.out.println("L'enter ha estat " + a + ".");  
    System.out.println("Llegiu un altre enter pel teclat:");  
    a = llegirEnterTeclat();  
    System.out.println("L'altre enter ha estat " + a + ".");  
}  
//1. Quin tipus de valor genera? Un enter (int)  
public int llegirEnterTeclat() {  
//2. Es fa el codi que llegeix un únic enter del teclat, com s'ha fet sempre  
//No canvia absolutament res...  
    Scanner lector = new Scanner(System.in);  
    int enterLlegit = 0;  
    boolean llegit = false;  
    while (!llegit) {  
        llegit = lector.hasNextInt();  
        if (llegit) {  
            enterLlegit = lector.nextInt();  
        }  
        else {  
            System.out.println("Això no és un enter.");  
            lector.next();  
        }  
    }  
    lector.nextLine();  
//3. Un cop fet, quina variable tindrà el resultat? "enterLlegit"  
//4. Cal fer "return" damunt seu  
    return enterLlegit;  
}
```

# Paràmetres de sortida

## Exemple

Revisem la funció main quan invoca la funció `llegirEnterTeclat()`:

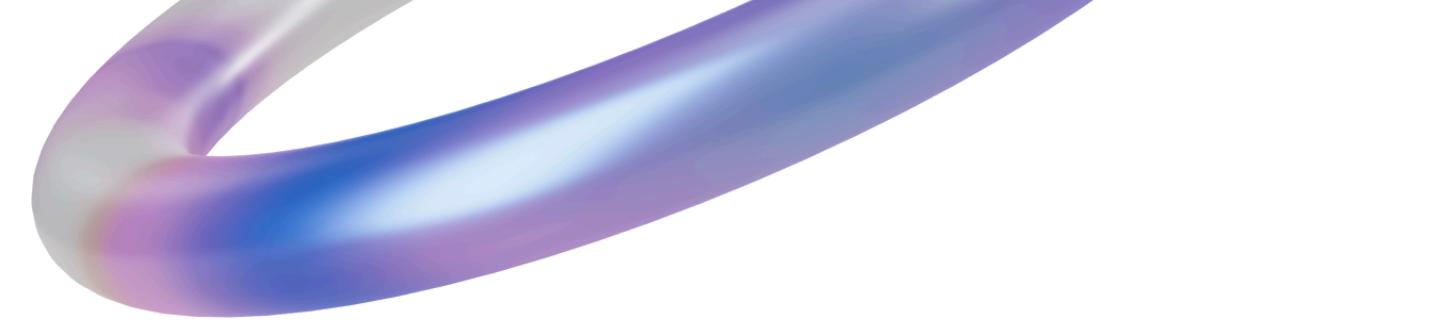
- **Recull el paràmetre de sortida?**
- **Quin és el nom de la variable que ha estat el paràmetre de sortida?**
- **Què fa amb el paràmetre de sortida el main?**

```
public static void main (String[] args) {  
    LlegirEnters programa = new LlegirEnters();  
    programa.inici();  
}  
public void inici() {  
    System.out.println("Llegiu un enter pel teclat:");  
    int a = llegirEnterTeclat();  
    System.out.println("L'enter ha estat " + a + ".");  
    System.out.println("Llegiu un altre enter pel teclat:");  
    a = llegirEnterTeclat();  
    System.out.println("L'altre enter ha estat " + a + ".");  
}
```

**Teniu el programa penjat a ALEXIA: Llegir Enters**

**Descarregeu-lo, compileu-lo i executeu-lo.**

# Exercici per entregar



## Llegir reals

> Modifiqueu l'exemple **Llegir reals** de manera que en lloc d'usar enters, funcioni amb reals (double). Assegureu-vos del seu correcte funcionament introduint nombres amb decimals.



# Gràcies

[mmoreno@pratfp.com](mailto:mmoreno@pratfp.com)

Desenvolupament d'aplicacions multiplataforma

**Prat**