Programació

UT4.4 Paràmetres

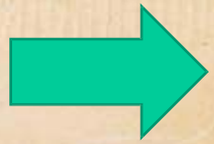
Paràmetrització de mètodes

- Un dels avantatges del disseny descendent era que cada subproblema era més fàcil d'abordar, i per tant, el problema general també.
- Un altre avantatge, és la **reusabilitat** de mètodes en un mateix programa.

Paràmetres

- Un paràmetre és un identificador dins de la descripció d'un procés (quan declarem un mètode). El valor pot variar en les diferents invocacions d'este procés.
- Tipus de paràmetres
  - D' **entrada**: Es proporcionen en la invocació del mètode.
  - D' **eixida**: Son retornats al finalizar el mètode.

Paràmetre d'entrada



- És un valor que s'estableix immediatament abans de començar el procés que estableix un mètode, de manera que indica les dades que va a tractar o bé que modifiquen el seu comportament.

- Es tractarà com una variables local interna al mètode.

- Eixa variable contindrà una copia del valor (argument) que hem indicat en la invocació del mètode.

Exemple de recepta
Procés: **Fregir**

Fregir carlotes

- Preparar paella per a fregir
- Agafar les carlotes
- Rossejar les carlotes
- Treure les carlotes de la paella
- Deixar les carlotes en un plat
- Netejar la paella

Fregir cebes

- Preparar paella per a fregir
- Agafar les cebes
- Rossejar les cebes
- Treure les cebes de la paella
- Deixar les cebes en un plat
- Netejar la paella

Aliment

Diferències a un mateix procés

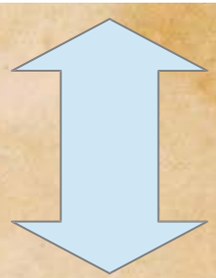
Declaració de mètodes amb paràmetres d'entrada

- No hi ha límit de paràmetres
- És **recomanable** no excedir-se en la seua declaració per a evitar errors i millorar la llegibilitat del programa
- IMPORTANT: **Els paràmetres es tracten com variables locals en eixe mètode**

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

Paràmetres d'entrada

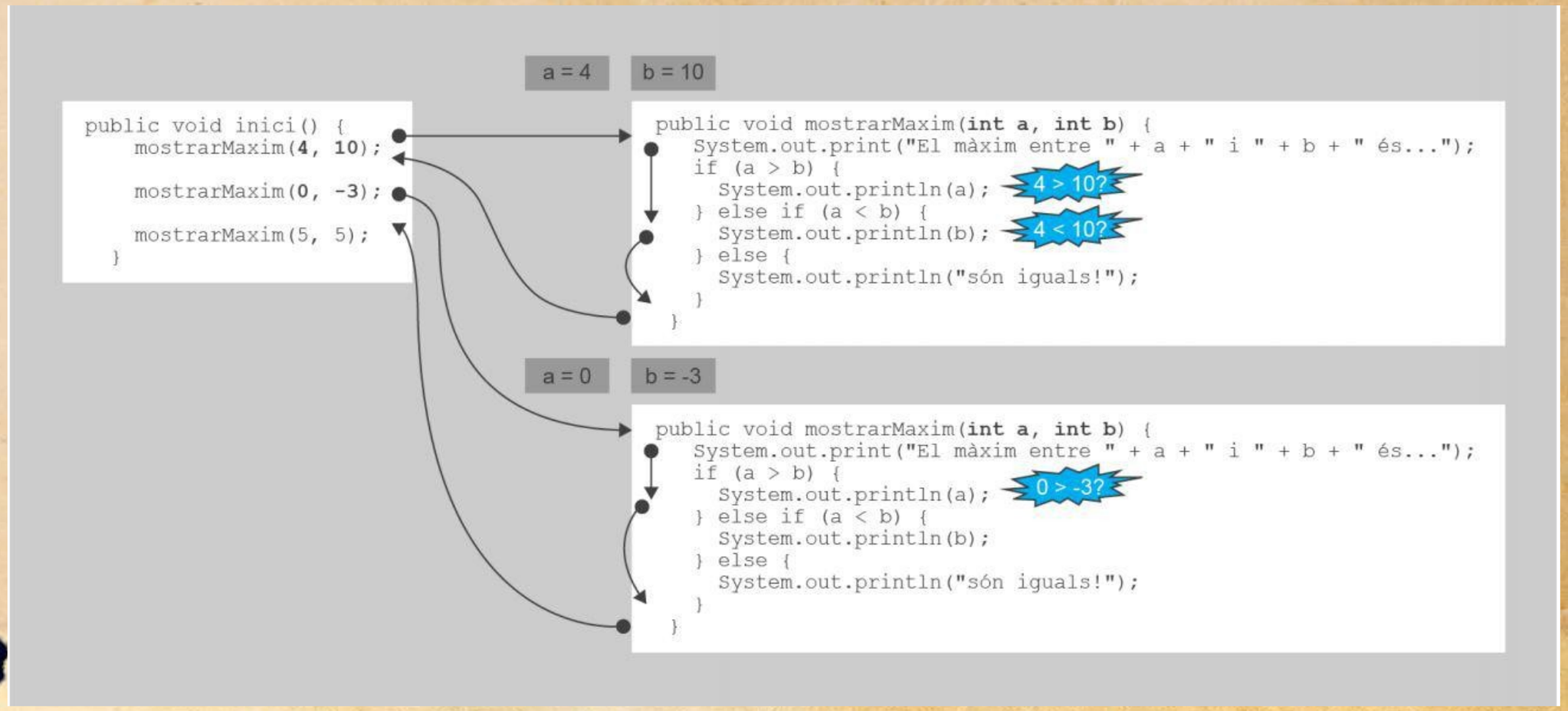
```
//El mètode mostrarMaxim té dos paràmetres d'entrada, de tipus enter
  public void mostrarMaxim(int a, int b) {
     ...
  }
```



```
public void mostrarMaxim() {
    //S'inicialitzen amb un valor inicial d'una manera especial que ja veureu
        aviat
    int a;
    int b;
    ...
  }
```

Invocació de mètodes amb paràmetres d'entrada

```
nomMetode(valor1, valor2, etc.);
```



Paràmetres d'entrada

- Encara que a l'exemple anterior s'han utilitzat literals per a invocar el mètode, es poden, per suposat, fer ús de variables o expressions (o altres mètodes). En eixe cas s'assigna (còpia) al paràmetre del mètode el contingut d'eixa variable (o el resultat d'una expressió o mètode).

Paràmetres d'entrada. Exemple

```
public class MaximParametreEntrada {
  public static void main (String[] args) {
    MaximParametreEntrada programa = new MaximParametreEntrada();
    programa.inici();
  }
  public void inici() {
    //Usem literals
    mostrarMaxim(4, 10);
    //Usem variables
    int i = 0;
    int j = -3;
    mostrarMaxim(i, j);
    //Usem expressions, amb literals o variables
    mostrarMaxim(2 + 3, i + 8);
  }
  //La resta del codi és igual...
}
```

Pràctica 9

- Crea un programa que cride diferents vegades a un mètode amb un únic paràmetre d'entrada de tipus enter. Este mètode escriurà per pantalla tants símbols '*' com indique el valor del paràmetre.
- Per invocar al mètode pots utilitzar tant un literal enter, com una variable entera a la qual prèviament li hi hauràs assignat el valor que vols passar al mètode.

Manipulació dels paràmetres d'entrada

- Una volta comença a executar-se el mètode, les variables representades per els seus paràmetres poden ser usades com qualsevol variable declarada al començament del mètode. Fins i tot es pot vore modificat el seu valor inicial.
- SUPER Important: una volta invocat el mètode, **un paràmetre conté una còpia** de la variable usada a la invocació del mètode (una còpia de l'argument).

Manipulació de paràmetres. Exemple

```
//Modifiqueu el valor d'un paràmetre. Afecta a la variable original?
public class ModificaParàmetre {
  public static void main (String[] args) {
    ModificaParàmetre programa = new ModificaParàmetre();
    programa.inici();
  }
  public void inici() {
    int i = 10;
    System.out.println("Abans de cridar el mètode \"i\" val " + i);
    modificarParametre(i);
    System.out.println("Després de cridar el mètode \"i\" val " + i);
  }
  //Té un únic paràmetre d'entrada, de tipus enter
  public void modificarParametre(int a) {
    //Ara hi ha una variable "a" declarada.
    //El seu valor depèn de com s'ha invocat el mètode.
    a = 0;
    System.out.println("Heu modificat el valor a " + a);
  }
}
```

Pràctica 10

- Realitza un programa similar l'anterior, però en este cas, el paràmetre serà un array d'enters. Dins del mètode, modifica un dada de l'array. El programa ha de mostrar les dades de l'array, abans i després d'invocar el mètode.
- Raona perquè ara s'obté esta eixida.

Pràctica 11

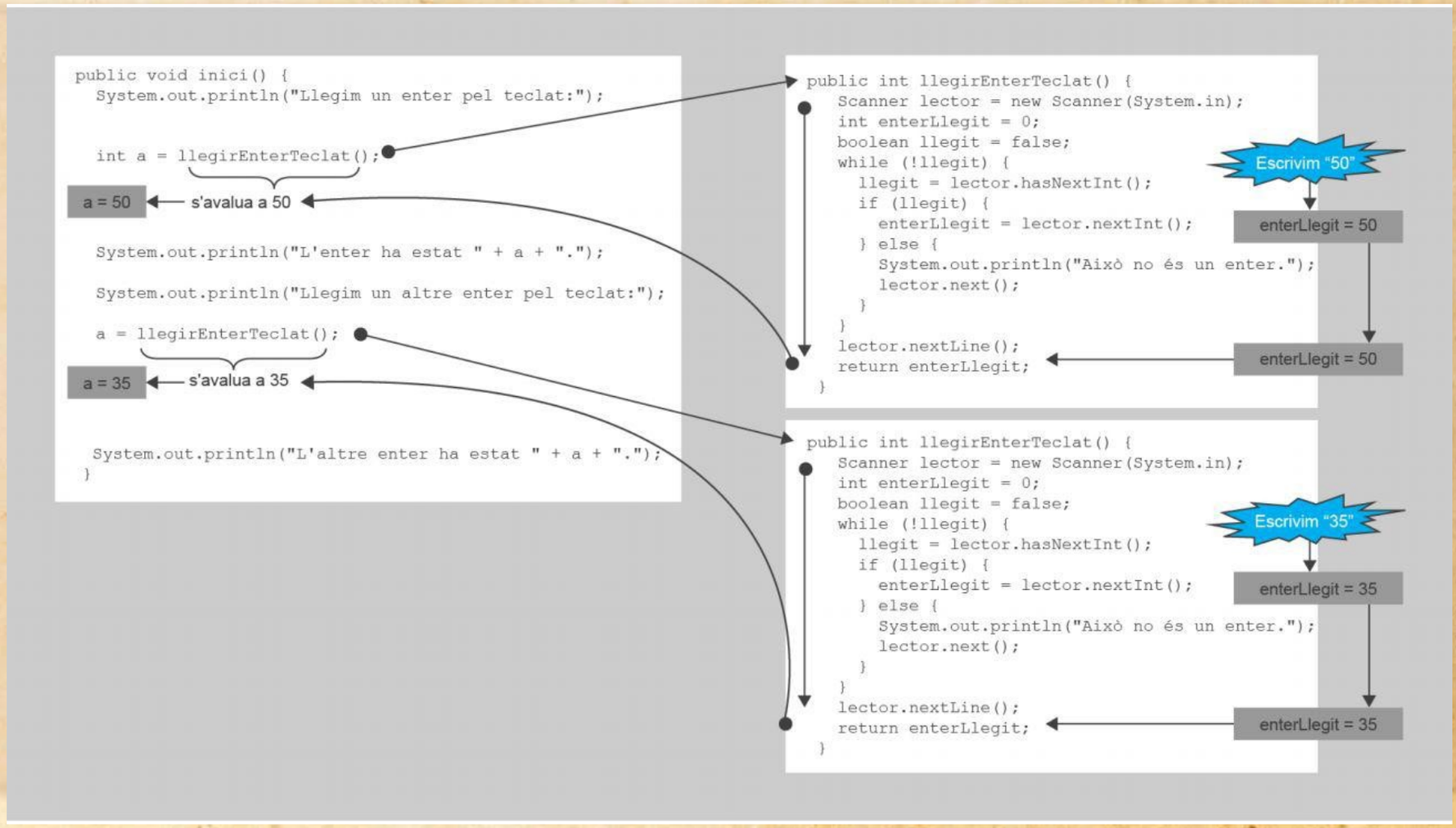
- Realitza un programa similar l'anterior, però en este cas, el paràmetre serà de tipus String.
- Es comporta com una variable de tipus bàsic?

Paràmetres d'eixida

- Indiquen els resultats obtinguts després de realizar un procès determinat.
- Per exemple, a l'exemple de la recepta, el paràmetre d'eixida seria el plat ja elaborat.
- En Java només es pot establir **un paràmetre d'eixida**, a través del seu tipus de dada.

Com convertir el mètode en una expressió, ja que la invocació equivaldrà al resultat que retorne.

Invocació d'un mètode amb paràmetres d'eixida



Paràmetres d'eixida (valor de retorn)

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

```
//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 té el resultat? "enterLlegit"
  //4. Cal fer "return" damunt seu
  return enterLlegit;
}
```

- Es farà ús de la paraula reservada "return" per a indicar el **valor de retorn**

Pràctica 12

- Modifica l’exemple anterior, per a que el paràmetre d’eixida siga de tipus real. Assegura't que el funcionament és correcte, introduint dades de tipus real per teclat.

Ús de paràmetres o variables globals?

- Fixeu-vos en este exemple

```
//Són variables globals
i = 4;
j = 8;
k = 12;
...
unMetode();
```

Suposant que dins de "unMetode()" es calcula un dada:

- A quines variables accedeix el mètode "unMetode()"?
- Quin és el resultat del mètode en este cas?
- Si es vol mostrar el resultat, a quina variable es troba?

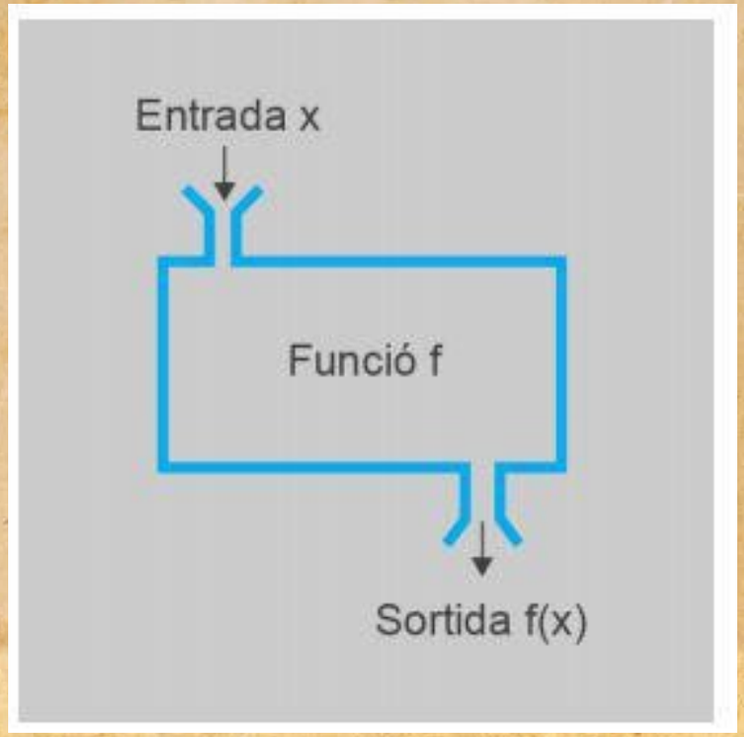
Paràmetres vs Globals

```
//No són variables globals
int i = 4;
int j = 8;
int k = 12;
...
k = unMetode(i, j);
```

- Totes les preguntes anteriors poden ser respostes en este cas.
- L'ús de paràmetres **millora la llegibilitat** del codi. Elimina interdependència i el fa més reutilitzable.

Principi d'ocultació

- Es defineix com la segregació dels elements en que es descompon un problema general, de manera que es separa la seua interfície contractual de la seua implementació.



Exemples

- El màxim (o el mínim) entre un conjunt de valors (array)
- La mitjana aritmètica dels valors dins d'un array
- Donada una nota, transformar-la a text (Aprovat o Suspès)
- Saber els dies que té un mes
- Buscar un valor concret en un array

Pràctica 13

- Efectua tots els programes indicats anteriorment fent ús de mètodes que utilitzen paràmetres d'entrada i d'eixida.

- Mostra l'entrada i eixida de cada un de ells per a comprovar el seu correcte funcionament.

Pràctica 14

- Fes un programa que demane un nombre enter per teclat. Donat un conjunt de valors dins d'un array, ha de calcular quantes voltes apareix el número introduït.
- Realitza mètodes tant per a preguntar el número, com per a buscar les seues ocurrències. L'array ha de ser declarat i inicialitzat en el mètode inici( ).
- **No es podran utilitzar variables globals.**

Pràctica 15

- Modifica el programa realitzat mitjançant el disseny descendent (temperatures), per a que ara deixe d'utilitzar variables globals (al menys l’array).