PROGRAMACIÓ ORIENTADA A OBJECTES
TRIMESTRE 2 – CURS 2021/2022
PRÀCTICA 3. SESSIÓ 1

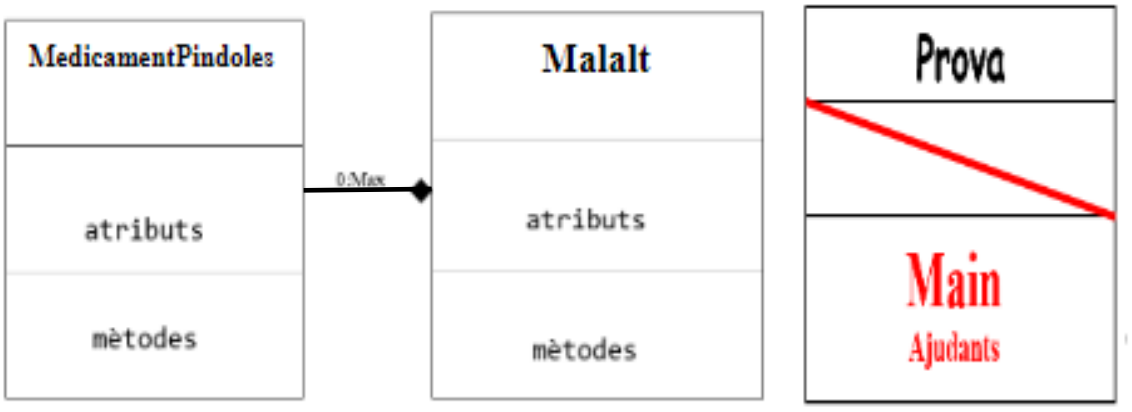
Objectiu: La classe **Object. Redefinició de mètodes**. La classe String. Objectes que són magatzems.
Durada: Dues sessions.
Lliurament: Llistat imprès dels fonts i penjar el projecte al Moodle.
Data Lliurament: Abans de la Pràctica 4.
Important: **Imprescindible per aprovar la pràctica** seguir les pautes de l'enunciat.

Treballarem al llarg de les dues sessions el mateix enunciat. Així com s'avanci en les sessions s'haurà d'ampliar i fer modificacions respecte el treballat a la sessió prèvia. Per tant, és imprescindible finalitzar el demanat en una sessió per poder continuar en la següent.

Enunciat

Ens demanen ajut per escriure un programa per un treball de recerca de batxillerat sobre les persones que prenen un format concret de medicament, aquells que es presenten en píndoles.

El programa està format per tres classes, una de ***Prova*** on hi ha el punt d'entrada (**`main`**), la classe ***Malalt***, i per últim la classe ***MedicamentPindoles***.



Important: els noms dels atributs i mètodes que estan explicitats en l'enunciat no es poden canviar. Els noms genèrics sí que els heu de canviar i donar-los el nom més adient. No podeu afegir ni atributs ni mètodes públics en cap de les classes i cal respectar la visibilitat indicada. Si no s'indica el contrari els atributs han de ser privats i els mètodes públics.

Primer escriu la classe MedicamentPindoles

Membres de la classe:

- Un atribut de tipus cadena amb el nom del medicament. Aquest nom s'ha d'emmagatzemar en **minúscules a excepció del primer caràcter que ha de ser en majúscules,** independentment de com el proporcioni l'usuari en la construcció de l'objecte. Useu mètodes de la classe String per obtenir el demanat. Exemples: **E**nalapril, **I**buprofeno.
- Un atribut de tipus enter amb el número d'unitats que hi ha quan compres el medicament. Aquest atribut a l'igual que el precedent no s'ha de poder modificar un cop assignats els corresponents valors, per tant, és imprescindible que siguin **constants**.
- Un atribut de tipus enter amb el número d'unitats preses (inicialment zero).
- El **constructor** que tindrà com a paràmetres el nom i el número d'unitats inicials. En el moment de la construcció el medicament "està sencer" (conté totes les píndoles).
- Un mètode funció que retorna un enter amb el número d'unitats que queden en el medicament: `public int quantesUnitatsQueden()`
- Un mètode per simular que es pren una píndola. Si encara queden píndoles, en pren una incrementant en una unitat l'atribut píndoles preses. Altrament no fa res: `public void prendrePindola()`
- Un mètode funció que retorna el número de píndoles preses:
  `public int getPindolesPreses()`
- Una funció que retorna el nom del medicament: `public String getNom()`
- **Redefineix el mètode equals,** un medicament és igual a un altre si tenen el mateix nom.
- **Redefineix el mètode toString** que ha de retornar una representació de l'objecte indicant: nom del medicament i el número de píndoles que li queden del total que tenia quan era sencer. El nom del medicament s'ha de visualitzar en majúscules.

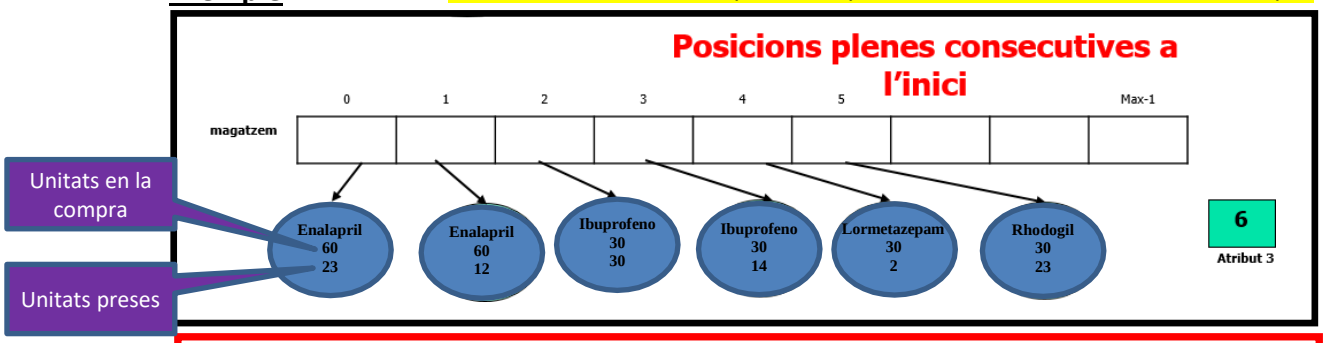
  Exemple:

  Nom del medicament IBUPROFENO amb 12 píndoles de 30

Segon escriu la classe Malalt

Un objecte Malalt és un magatzem de medicaments. El magatzem és un vector d'objectes ***MedicamentPindoles***. **Aquest magatzem ha d'estar ordenat ascendentment en funció del nom del medicament**. A igualtat de nom s'han d'ordenar **descendentment** pel número de píndoles que té l'envàs quan es compra, i a igualtat d'aquest valor cal ordenar-ho **descendentment** en funció del número de píndoles presses. Així doncs tots els medicaments que tenen el mateix estaran ubicats consecutivament. La relació d'ordre establerta entre objectes String és alfabètica (no pas per longitud).

Exemple Un mateix medicament pot estar present en diferents unitats de compra

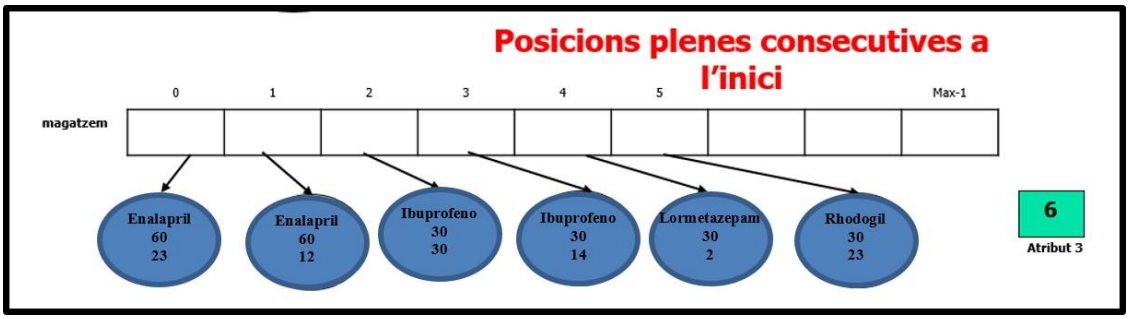


Important. Quan apliqueu mètodes de cerca no necessàriament s'ha de continuar fins al final, al tractar un element que no és el buscat però és major al buscat ja no cal continuar. Cal usar el mètode `compareTo` de la classe String.

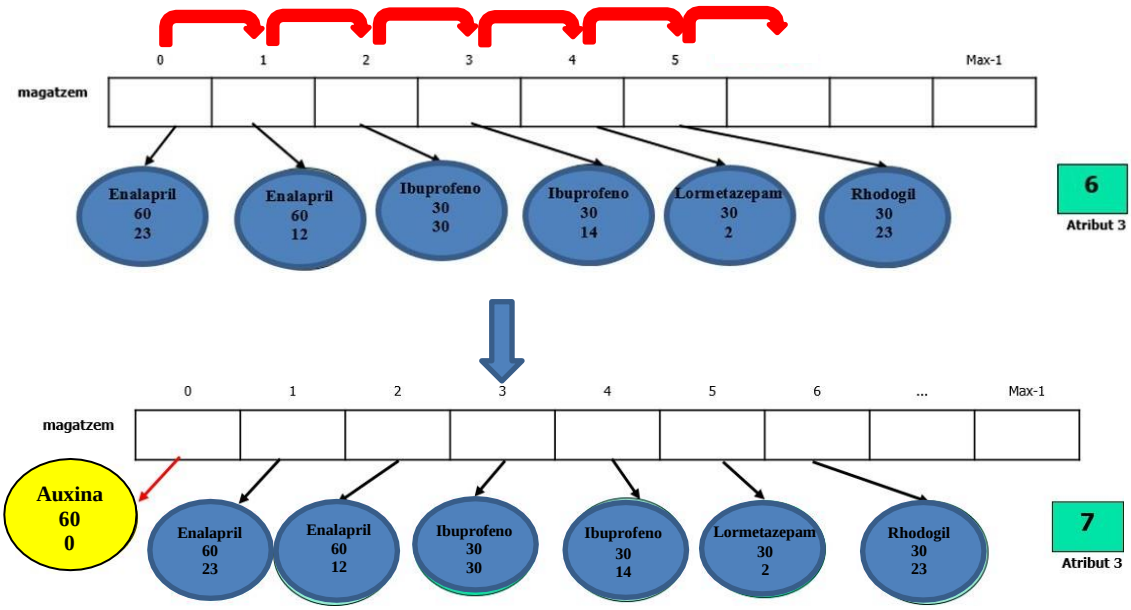
Membres de la classe:

- Un primer **atribut 1**, per emmagatzema el nom del malalt.
- Un **atribut 2** magatzem, ha de ser un vector amb els medicaments que té el malalt. **Totes les posicions plenes han d'estar consecutives**. Aquest atribut és un magatzem d'objectes MedicamentPindoles.
- Un **atribut 3** numèric que emmagatzema el número de medicaments que té el malalt. Per tant tindrà la dimensió real de l'atribut vector anterior.
- Un darrer **atribut 4 constant** que emmagatzema les unitats d'increment del magatzem quan està totalment ple (consultar transparències teoria: Magatzems).
- Mètode **constructor** que rep com a paràmetres el nom del malalt, el número màxim de medicaments del magatzem i les unitats d'increment del vector que cal augmentar quan sigui necessari. De moment el magatzem està buit.
- Mètodes consultors **`getXXX()`** pels atributs 1, 3 i 4. El magatzem no es pot consultar.
- Mètode per consultar **un** objecte del magatzem. El mètode rep un enter corresponent a la ubicació dins del magatzem de l'objecte a consultar. Si el valor del paràmetre no és correcte, el mètode retorna la referencia nul·la.

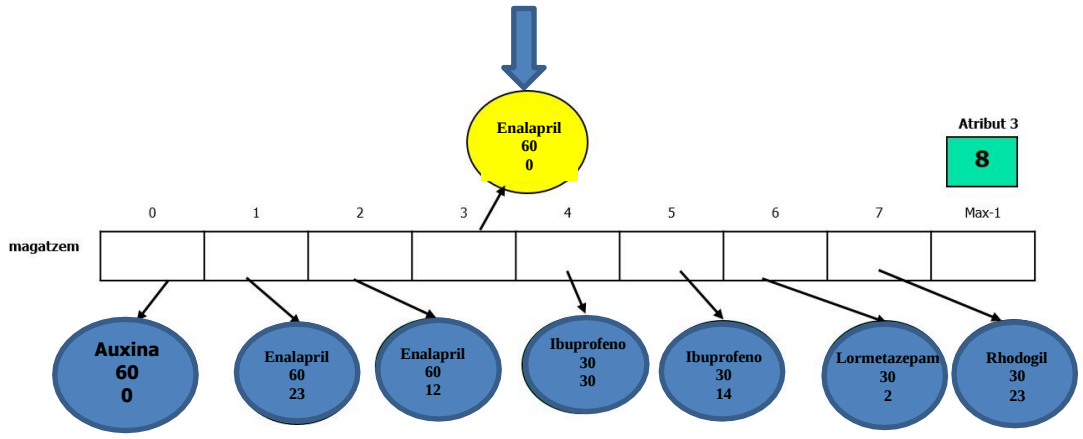
  `MedicamentPindoles getMedicamentPindoles(int quin)`
- Mètode **`MedicamentPindoles getMedicamentPindolesNoBuit()`** que retorna el primer medicament del vector que no és buit, és a dir que conté com a mínim una píndola. Si no hi ha cap retorna una referència nul·la.
- Mètode **`void comprarMedicamentPindoles(MedicamentPindoles p)`** que simula la compra **d'un** medicament. Si el medicament ***p*** hi cap al vector li afegeix, altrament primer haurà **d'ampliar** el magatzem per donar-li cabuda. Les unitats en que s'augmenta el magatzem han de ser les unitats indicades al darrer atribut descrit. **El magatzem té els objectes ordenats per tant després de la inserció hi ha de continuar estant** (no necessàriament s'ha d'afegir al final).



Adquirir el medicament de nom **Auxina**
Com el nom és més petit que tots els que hi ha al magatzem s'ha d'afegir a la component d'índex 0. **S'ha de fer lloc!**



Adquirir el medicament de nom **Enalapril de 60 unitats**
S'ha de buscar el lloc que li pertoca dins del magatzem. En aquest cas s'ha d'ubicar a la component d`índex 3. **S'ha de fer lloc!**



- **Sobrecarrega** del mètode previ, **`void comprarMedicamentPindoles(String p, int pindoles)`** que simula l'adquisició d'un medicament, però en aquest cas no es dona l'objecte MedicamentPindoles, arriben dos paràmetres corresponents al nom i quantitat de píndoles de la caixa, per tant s'ha de construir prèviament a la inserció. **Aquest mètode obligatòriament ha d'invocar al precedent**.
- Mètode **`int totalPindolesQueden()`** que retorna el total de píndoles que li queden al malalt de **tots** els medicaments del magatzem.
- Sobrecarrega del mètode precedent, **`int totalPindolesPreses(String nom)`** que són del nom indicat pel paràmetre**. Imprescindible invocar al mètode `equals` de la classe MedicamentPindoles (no pas de la classe String) per calcular el valor a retornar.**
- Mètode **`int maximPindoles()`** que ha de localitzar el medicament del magatzem que té més píndoles a la caixa, en retorna quantes en té (en cas de coincidència irrellevant el valor retornat).
- El mètode **`String numMedicamentsPerQueden()`** que ha de retornar dades envers els medicaments de l'objecte Malalt. Concretament ha de calcular quants medicaments hi ha al magatzem amb **una** píndola (li queda 1 píndola), quants hi ha amb **dues** píndoles (li queden 2 píndoles), quants amb **tres** Aquestes dades les ha de retornar en format cadena.

  Per realitzar aquest càlcul **només es pot fer un recorregut del magatzem**, en aquest **únic** recorregut es recolliran en dos vectors totes les dades necessàries pel càlcul demanat. Un vector emmagatzemarà el número de píndoles que queden, en direm vector ***queden*** i a la mateixa component de l'altre vector, en direm ***quants,*** el número de medicaments que hi ha al magatzem amb aquella quantitat de píndoles.

  Exemple:

  Vector *queden*

0	1	2	3	4	5	.....	....	.....	n
12	10	6	8	9					

0	1	2	3	4	5	.....	....	.....	n
5	3	2	1	1					

  Vector *quants*

  Aquestes dades indiquen que amb 12 píndoles hi ha 5 medicaments, amb 10 píndoles 3 medicaments ...

Per implementar aquest mètode cal escriure **dos mètodes estàtics i privats** que en el seu ús donaran solució al demanat:

- `private static int troba(MedicamentPindoles caixa, int[] queden, int quants)`
- `private static String crear(int[] queden, int[] quants, int quantes)`

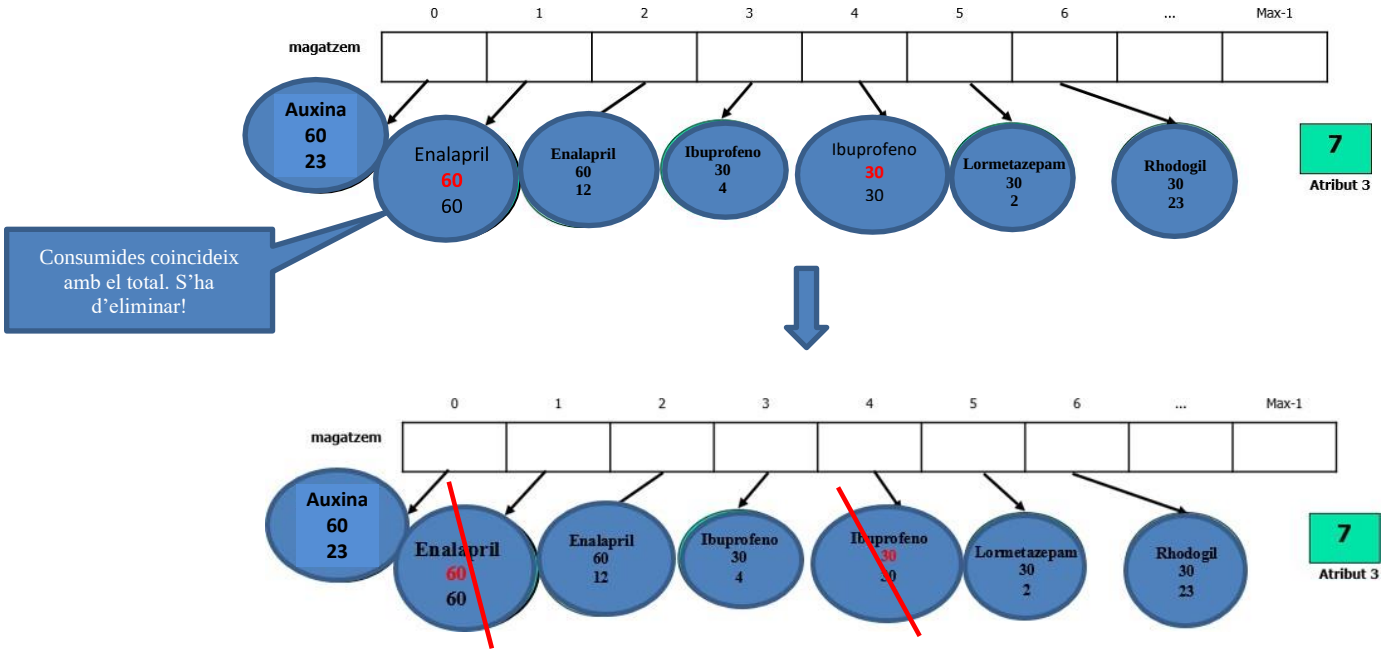
El mètode **troba** retorna l'índex de la posició del vector **queden** on es ***troba*** la quantitat de píndoles que li queden al **medicament** donat en el primer paràmetre. El segon paràmetre és el vector on buscar i el darrer el nombre de posicions plenes d'aquest vector. Si al vector no hi és el valor numèric buscat ha de retornar el numèric -1.

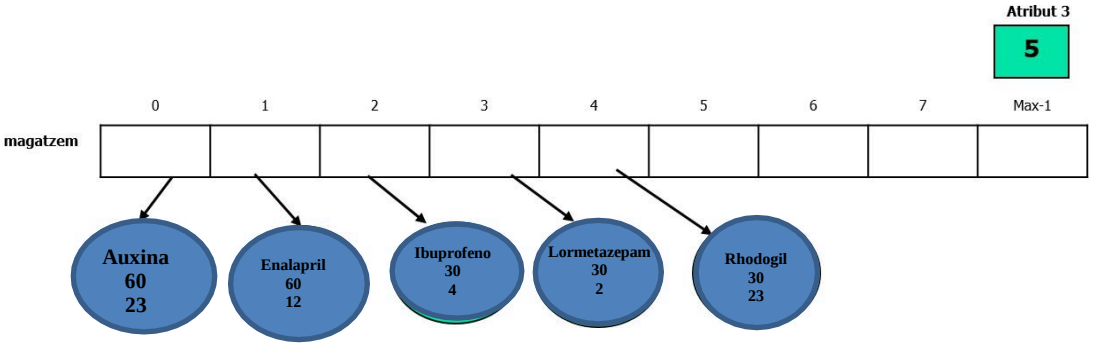
El mètode **crear** seguint un esquema de **recorregut** crea la cadena a retornar a partir de les taules ***queden*** i ***quants*** amb ***quantes*** posicions plenes. Per cada component cal crear el text: "Amb X píndoles queden M medicaments" on X són el nombre de píndoles que hi ha i M el número de medicaments que en tenen X. El text de cada component ha d'estar separat del següent pel caràcter guió. Per l'exemple previ seria:

"Amb 12 píndoles queden 5 medicaments - Amb 10 píndoles queden 3 medicaments – Amb 6 píndoles queden 2 medicaments" ....

- Un mètode `int eliminarMedicamentPindoles()` per eliminar del magatzem tots els medicament que tenen 0 píndoles. Totes les posicions buides estan i han de quedar consecutives al final de la taula. Recordeu d'actualitzar l'atribut corresponent al número de medicaments que té el malalt. El magatzem un cop feta l'eliminació ha de continuar estant ordenat. El mètode ha de retornar el nombre de medicaments que s'han eliminat del magatzem.

Exemple:





- Mètode **`MedicamentPindoles[] donaMedicamentsBuits()`** que ha de retornar dins d'un vector, que ha d'estar totalment ple, totes les referències als medicaments del magatzem que estan buits. A diferència del mètode anterior, aquest no modifica cap atribut de l'objecte. Si no en hi ha cap retorna una referència nul·la.

- **Redefineix** el mètode **`String toString()`** per a que generi una cadena amb un llistat de tots els medicaments que té el magatzem. Per fer aquest tractament invoqueu **obligatòriament** al mètode `toString` de la classe MedicamentPindoles per a cada objecte del magatzem.

  Exemple:

  ```
  1. Nom del medicament AUXINA amb 23 píndoles de 60
  2. Nom del medicament ENALAPRIL amb 12 píndoles de 40
  3. Nom del medicament IBUPROFENO amb 10 píndoles de 30
  ```

- **Redefineix** el mètode **`equals`** per la comparació amb igualtat d'objectes Malalt. Dos malalts es consideren iguals si tenen el mateix nombre de medicaments amb la mateixa quantitat de píndoles (independent del nom del medicament). Per implementar aquest mètode procediu de la següent manera:
  - 1.- Invoqueu al mètode **`String numMedicamentsPerQueden()`** per a cadascun dels malalts a comparar. Com a resultat obteniu un objecte String per cada malalt.
  - 2.- **Trenqueu** cada String obtingut en una taula de components String on cada component serà una quantitat de píndoles amb el seu corresponent numero d'unitats. Per això utilitzareu de la classe `String` el mètode `split()` posant com a paràmetre el caràcter separador, el guió.

    Exemple: "Amb 12 píndoles queden 5 medicaments - Amb 10 píndoles queden 3 medicaments – Amb 6 píndoles queden 2 medicaments"

    Mètode **`split`**

0	1	2
Amb 12 píndoles queden 5 medicaments	Amb 10 píndoles queden 3 medicaments	Amb 6 píndoles queden 2 medicaments

  - 3.- **Ordeneu ascendentment** les taules usant un mètode d'ordenació de taules de la classe Arrays. El mètode **sort** a utilitzar és un **mètode estàtic** de la classe. Serà necessari fer la importació del paquet **java.util**

0	1	2
Amb 6 píndoles queden 2 medicaments	Amb 10 píndoles queden 3 medicaments	Amb 12 píndoles queden 5 medicaments

- 4.- **Convertiu les taules a String i compareu-los**, useu mètodes de la classe String.
  "Amb 6 píndoles queden 2 medicamentsAmb 10 píndoles queden 3 medicamentsAmb 12 píndolesqueden 5 medicaments"

Reflexioneu: els següents Strings corresponents a dues cadenes de malats que **són iguals**, tot i què el mètode `String numMedicamentsPerQueden()` no retorna el mateix:

Amb 10 píndoles queden 3 medicaments - Amb 6 píndoles queden 2 medicaments - Amb 12 píndoles queden 5 medicaments

Amb 6 píndoles queden 2 medicaments - Amb 10 píndoles queden 3 medicaments - Amb 12 píndoles queden 5 medicaments

Tercer escriu la classe Prova

Amb l'objectiu de provar el correcte funcionament d'una part del codi desenvolupat, escriu el main i ajudants descrits a continuació. La classe Prova no pot tenir cap atribut de cap mena!

Mètode `main` que ha de:

- 1.- Crear una malalta, la Maria Fernández que comprarà com a màxim 10 medicaments, ara adquireix 2 medicaments de "Enalapril amb 60 píndoles" i un medicament "Ibuprofeno amb 20".
- 2.- Crear un malalt, en Joan Moll que adquirirà com a màxim 15 medicaments, adquireix ara caixes de Auxina "dues amb 30 píndoles i dues de 35" i 2 caixes de "Ibuprofeno amb 20".
- 3.- Mostrar les dades de cadascun dels malalts, primer un llistat del que ha comprat i a continuació detallat per nombre de medicaments i unitats.
- 4.- Fer que cada malalt es prengui de tots els medicaments no buits totes les píndoles.
- 5.- Tornar a mostrar les dades de cadascun dels malalts.

Per implementar el punt 4 implementa i usa els **procediments** següents:

- `private static void prendreTotsElsMedicamentsComprats(Malalt m)` Cal obtenir tots els medicaments no buits del malalt donat en el paràmetre del procediment, i per cadascun d'ells se'l prengui tot **mitjançant** el procediment `prendreDelMedicament(MedicamntPindoles)` que heu d'implementar a continuació.
- `private static void prendreDelMedicament(MedicamentPindoles m)` per fer-ho primerament cal obtenir quantes píndoles queden, i després se les

prendrà d'una a una **mitjançant** el mètode **`prendrePindola()`** de la classe MedicamentPindoles**.**

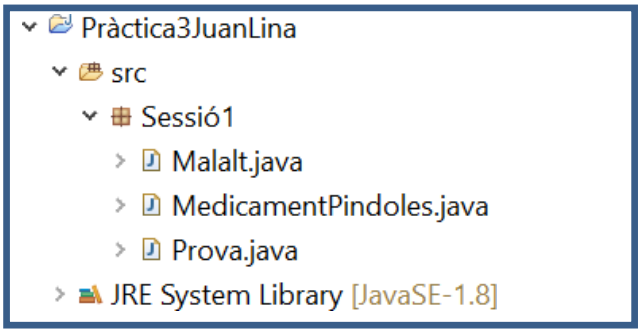
Exemple d'execució

```
El malalt Maria Fernández té:
*****************************
 1. Nom del medicament ENALAPRIL amb 60 píndoles de 60
 2. Nom del medicament ENALALPRIL amb 60 píndoles de 60
 3. Nom del medicament IBUPROFENO amb 20 píndoles de 20

Llistat per unitats
*******************
 1. Amb 60 píndoles queden 2 medicaments
 2. Amb 20 píndoles queden 1 medicaments

El malalt Joan Moll té:
***********************
 1. Nom del medicament AUXINA amb 30 píndoles de 30
 2. Nom del medicament AUXINA amb 30 píndoles de 30
 3. Nom del medicament AUXINA amb 35 píndoles de 35
 4. Nom del medicament AUXINA amb 35 píndoles de 35
 5. Nom del medicament IBUPROFENO amb 20 píndoles de 20
 6. Nom del medicament IBUPROFENO amb 20 píndoles de 20

Llistat per unitats
*******************
1. Amb 30 píndoles queden 2 medicaments
2. Amb 35 píndoles queden 2 medicaments
3. Amb 20 píndoles queden 2 medicaments

Els malalts es prenen totes les píndoles!!!!!

El Maria Fernández es pren totes les píndoles que té
***************************************************
El malalt Maria Fernández té:
 1. Nom del medicament ENALAPRIL amb 0 píndoles de 60
 2. Nom del medicament ENALAPRIL amb 0 píndoles de 60
 3. Nom del medicament IBUPROFENO amb 0 píndoles de 20

Llistat per unitats
*******************
1. Amb 0 píndoles queden 3 medicaments

El Joan Moll es pren totes les píndoles que té
**********************************************
El malalt Joan Moll té:
 1. Nom del medicament AUXINA amb 0 píndoles de 30
 2. Nom del medicament AUXINA amb 0 píndoles de 30
 3. Nom del medicament AUXINA amb 0 píndoles de 35
 4. Nom del medicament AUXINA amb 0 píndoles de 35
 5. Nom del medicament IBUPROFENO amb 0 píndoles de 20
 6. Nom del medicament IBUPROFENO amb 0 píndoles de 20

Llistat per unitats
*******************
1. Amb 0 píndoles queden 6 medicaments
```

observeu que el String que retorna el mètode s'ha de trencar

Organització. Totes les pràctiques s'han de fer en grups de 2 estudiants

Seguireu la mateixa estructura que la usada a les pràctiques precedents. Cal crear un nou projecte amb nom **Pràctica3Cognom1Nom1&&Cognom2Nom2**. El projecte ha de tenir un paquet per cada sessió. En aquesta sessió hi ubicareu les tres classes demanades.



Què se us subministra?

L'enunciat amb les especificacions i línies a seguir per el desenvolupament del disseny de classes.

Què s'ha de lliurar i com?

S'ha de lliurar la carpeta que conté el projecte Eclipse amb el vostre desenvolupament de la pràctica. La carpeta s'ha de lliurar amb tot el seu contingut i comprimida amb ZIP o RAR.

També s'ha de lliurar un **llistat en paper** del codi desenvolupat (no la classe Prova). El format de lliurament d'aquest codi ha de seguir el patró indicat en la presentació de l'assignatura: amb portada, índex, número de pàgina, tabulació ... En **aquest llistat** cal que indiqueu:

- **la distribució de la feina entre els dos estudiants. És a dir, el grau de participació de cada membre del grup en la realització d'aquesta activitat.**
- **Si el programa no funciona cal que indiqueu quina/es parts no funcionen explicant que és el que passa.**

On s'ha de lliurar?

El lliurament del projecte es farà a través de la plataforma Moodle i no s'acceptarà cap altra via. Feu atenció a la data i hora límit.

El lliurament en paper es farà directament a la professora a l'inici de la primera sessió de la pràctica 4.

Quan s'ha de lliurar?

El lliurament es podrà fer fins el **dia indicat a sota**. Tingueu present que a partir d'aquesta hora el sistema bloquejarà, de manera automàtica, la possibilitat de lliurament.

Lliurament Moodle → 10 Març a les 23:50h
Lliurament en paper → 11 de Març inici sessió de pràctiques

Pauta de correcció:

En aquesta pràctica de dues sessions resoldreu un únic enunciat que s'anirà ampliant sessió a sessió.

Aquesta sessió 1 de la pràctica 3 té un pes del 70% de la qualificació total de la pràctica 3.

Valoració (sobre 10 punts):

- 1.- Escriptura de la classe **Malalt** (**8 punts**)
- 2.- Escriptura de la classe **MedicamentPindoles** (**1 punt**)
- 2.- Escriptura de la classe **Prova** (**1 punt**)

La correcció valorarà:

- 1.- **Descomposició funcional** aplicada, tant per un reaprofitament del codi com per la claredat de la solució mostrada. És imprescindible que cada tasca amb una funcionalitat pròpia tingui el seu propi procediment.
- 2.- L'aplicació dels **esquemes** adients (recorregut i cerca). El magatzem està ordenat, per tant tingueu-ho en compte a l'aplicar **l'esquema de cerca**
- 3.- La **robustesa**, és a dir el control de què les dades demanades a l'usuari siguin correctes.
- 4.- **Eficiència** en quan a no fer càlculs repetits amb unes mateixes dades.

La qualificació dependrà dels ítems indicats prèviament. **Obligatòriament heu de seguir les pautes de programació** donades en l'escriptura del programa. La vostra solució s'ha d'ajustar a l'enunciat de l'exercici.