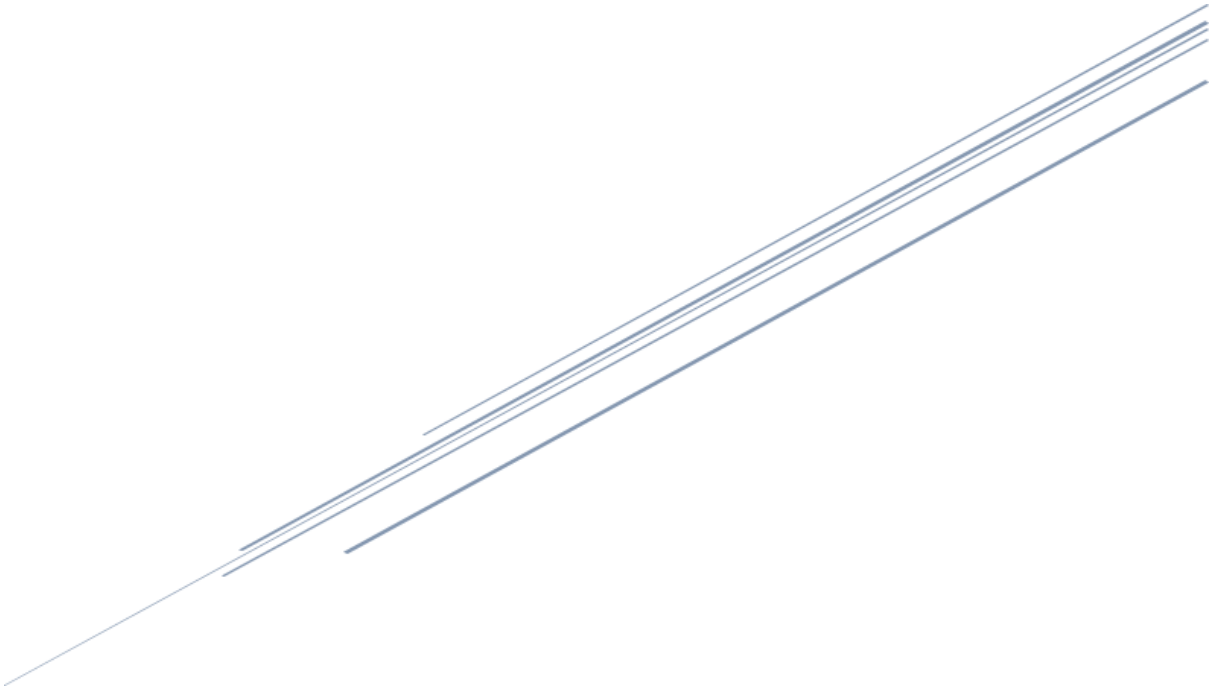
Pràctica 3



Rems Nalivaiko

Pol Rubio

13 de mar. de 2022

103112 - PROGRAMACIÓ ORIENTADA OBJECTE

Fundació TecnoCampus Mataró-Maresme

Índex:

Sessió 1: **3**

Sessió 2: **13**

Organització de la feina: **26**

Valoració: **26**

Sessió 1:

Es tracta de fer un programa que amb un tauler de dimensions determinades i donat un estat inicial vagi aplicant les regles anteriors:

Codi:

Malalt.java

```
package Sessio1;

import java.util.Arrays;

public class Malalt {
	private String nom;
	private MedicamentPindoles[] medicaments;
	private int num; // num. de medicaments
	private final int increment;


	// CONSTRUCTORs
	public Malalt(String nom, int numMax, int increment) {
		this.nom = nom;
		medicaments = new MedicamentPindoles[numMax];
		this.increment = increment;
	}


	public void comprarMedicamentPindoles(MedicamentPindoles p) {
		if(num==medicaments.length)
			ampliar();
		medicaments[num]=p; num++;

		ordenar();
	}
	public void comprarMedicamentPindoles(String p, int pindoles) {
		MedicamentPindoles nouMedicament = new MedicamentPindoles(p,
pindoles);
		comprarMedicamentPindoles(nouMedicament);
```

```
	}


	public int totalPindolesQueden() {
		int n=0;
		for(int i=0; i<num; i++) {
			n += medicaments[i].quantesUnitatsQueden();
		}
		return n;
	}

	public int totalPindolesPreses(String nom) {
		for(int i=0; i<num; i++) {
			if(medicaments[i].equals(nom)) return
medicaments[i].quantesUnitatsQueden();
		}
		return 0;
	}


	public int maximPindoles(){
		int n=0;
		for(int i=0; i<num; i++) {
			if(medicaments[i].quantesUnitatsQueden()>n)
				n=medicaments[i].quantesUnitatsQueden();
		}
		return n;
	}

	public String numMedicamentsPerQueden() {
		int queden[]=new int[num]; int quants[]=new int[num];
		int quantes=0; // posicions plenes a queden[]

		for(int i=0; i<num; i++) {
			int x=troba(medicaments[i], queden, quantes); int
posicio=x;

			if(x==-1 || quantes==0){
```

```
queden[quantes]=medicaments[i].quantesUnitatsQueden();
                        posicio=quantes; quantes++;
                  }
                  quants[posicio]++;
            }
            return crear(queden,quants,quantes);
      }

      public int eliminarMedicamentPindoles() {
            int quants=0; int eliminats=0;
            for(int i=0; i<num; i++) {
                  if(medicaments[i].quantesUnitatsQueden()==0){
                        medicaments[i]=null; eliminats++;
                  } else {
                        medicaments[quants]=medicaments[i]; quants++;
                  }
            }

            return eliminats;
      }

      public MedicamentPindoles[] donaMedicamentsBuits() {
            MedicamentPindoles[] buits_tmp=new MedicamentPindoles[num];
int cnt=0;
            for (MedicamentPindoles i : medicaments){
                  if (i==null) continue; // legal? si no ho es, negar el
if i que ho envolti tot.
                  if (i.quantesUnitatsQueden()==0){
                        buits_tmp[cnt]=i; cnt++;
                  }
            }

            MedicamentPindoles[] buits=new MedicamentPindoles[cnt];
            for (int x=0;x<cnt;x++) buits[x]=buits_tmp[x];
            return buits;
      }
```

```
	// OVERWRITEs
	public String toString() {
		String msg="";
		for (int x=0;x<num;x++)
			msg+=(x+1)+". "+medicaments[x].toString()+"\n";

		return msg;
	}
	public boolean equals(Malalt malaltB){
		// si tenen la mateixa quantitat de pindoles
		String[]
llista_strMalaltA=numMedicamentsPerQueden().split("-");
		String[]
llista_strMalaltB=malaltB.numMedicamentsPerQueden().split("-");

		Arrays.sort(llista_strMalaltA);
Arrays.sort(llista_strMalaltB);

		String strMalaltA=Arrays.toString(llista_strMalaltA);
		String strMalaltB=Arrays.toString(llista_strMalaltB);

		return strMalaltA.equals(strMalaltB);
	}


	//GETs
	public String getNom() {return nom;}
	public int getNum() {return num;}
	public int getIncrement() {return increment;}
	public MedicamentPindoles getMedicamentPindoles(int quin) {
		if(quin < num) return medicaments[quin];
		return null;
	}
	public MedicamentPindoles getMedicamentNoBuit() {
		for(int i=0; i<num; i++) {
```

```
                    if(medicaments[i].quantesUnitatsQueden()>0) return
medicaments[i];
              }
              return null;
       }


       //PRIVATEs
       private void ampliar() {
              MedicamentPindoles[] nousMedicaments = new
MedicamentPindoles[medicaments.length + increment];
              for(int i = 0; i<this.medicaments.length; i++)
nousMedicaments[i] = medicaments[i];
              medicaments = nousMedicaments;
       }
       private void ordenar() {
              MedicamentPindoles aux;
              for(int x=1; x<num; x++){
                    for(int y=num-1; y>x; y--){
                           int
comparacio=medicaments[y].getNom().compareTo(medicaments[y-1].getNom());

                           aux=medicaments[y];
                           int pindolesPreses1=0; int pindolesPreses2=0;
                           int pindolesMax1=0; int pindolesMax2=0;
                           if (comparacio==0) {

pindolesPreses1=medicaments[y].getPindolesPreses();

pindolesPreses2=medicaments[y-1].getPindolesPreses();


pindolesMax1=pindolesPreses1+medicaments[y].quantesUnitatsQueden();

pindolesMax2=pindolesPreses2+medicaments[y-1].quantesUnitatsQueden();
                           }

                           if (comparacio<0 || (comparacio==0 &&
```

```
(pindolesMax1>pindolesMax2 || (pindolesMax1==pindolesMax2 &&
pindolesPreses1>pindolesPreses2))){
                                    medicaments[y]=medicaments[y-1];
medicaments[y-1]=aux;
                              }
                        }
                  }
      }

      private static int troba(MedicamentPindoles caixa, int[] queden, int
quants) {
            int quantitat=caixa.quantesUnitatsQueden();
            for(int i=0; i<quants; i++) {
                  if (queden[i]==quantitat) return i;
            }
            return -1;
      }
      private static String crear(int[] queden, int[] quants, int quantes)
{
            String msg="";
            for(int i=0; i<quantes; i++)
                  msg+="Amb "+queden[i]+" píndoles queden "+quants[i]+"
medicaments - ";

            return msg;
      }
}
```

MedicamentPindoles.java

```
package Sessio1;

public class MedicamentPindoles {
      private final String nom;
      private final int unitats;
      private int preses=0;
```

```
	public MedicamentPindoles(String nom, int unitats){
		this.nom =
nom.substring(0,1).toUpperCase()+nom.substring(1).toLowerCase();
		this.unitats = unitats;
	}

	//GETs
	public int quantesUnitatsQueden() {return unitats-preses;}
	public void prendrePindola() {if(preses < unitats) preses++;}
	public int getPindolesPreses() {return preses;}
	public String getNom() {return nom;}

	//OVERRIDES
	public boolean equals(Object o) {
		MedicamentPindoles altreMedicament;

		if(o instanceof MedicamentPindoles) {
			altreMedicament = (MedicamentPindoles) o;
			return  this.getNom().equals(altreMedicament.getNom());
		}
		return false;
	}
	public boolean equals(String altre_nom) {
		return this.getNom().equals(altre_nom);
	}
	public String toString() {
		return "Nom del medicament " + nom + " amb " +
quantesUnitatsQueden() + " pindoles de " + unitats;
	}
}
```

Prova.java

```
package Sessio1;

public class Prova {
    public static void main(String args[]){
```

```
        int increment=2;
        Malalt malalts[] = {new Malalt("Maria Fernández", 10, increment),
new Malalt("Joan Moll", 15, increment)};

        malalts[0].comprarMedicamentPindoles(new
MedicamentPindoles("Enalapril",60));
        malalts[0].comprarMedicamentPindoles(new
MedicamentPindoles("Enalalpril",60));
        malalts[0].comprarMedicamentPindoles(new
MedicamentPindoles("Ibuprofeno",20));

        for (int x=0;x<2;x++) {
            malalts[1].comprarMedicamentPindoles(new
MedicamentPindoles("Auxina",30));
            malalts[1].comprarMedicamentPindoles(new
MedicamentPindoles("Auxina",35));
            malalts[1].comprarMedicamentPindoles(new
MedicamentPindoles("Ibuprofeno",20));
        }


        imprimirDades(malalts);

        for (Malalt malalt: malalts)
prendreTotsElsMedicamentsComprats(malalt); System.out.println("MEDS ARE
GONE\n\n");

        imprimirDades(malalts);
    }

    private static void prendreTotsElsMedicamentsComprats(Malalt m) {
      MedicamentPindoles med;
        do {
            med=m.getMedicamentNoBuit();
            if(med!=null) prendreDelMedicament(med);
        } while (med!=null);
    }
```

```
    private static void prendreDelMedicament(MedicamentPindoles m){
        for (int x=m.quantesUnitatsQueden(); x>0; x--) m.prendrePindola();
    }

    private static void imprimirDades(Malalt malalts[]){
        for (Malalt malalt: malalts){
            String msg="El malalt "+malalt.getNom()+" té:";
System.out.println(msg);
            for (int i=0; i<msg.length(); i++) System.out.print("*");
System.out.println("");
            System.out.println(malalt.toString());

            msg="Llista per unitats:"; System.out.println(msg);
            for (int i=0; i<msg.length(); i++) System.out.print("*");
System.out.println("");
            String[] parts=malalt.numMedicamentsPerQueden().split(" - ");
            for (int x=0; x<parts.length; x++)
                System.out.println(x+". "+parts[x]);

            System.out.println("\n");
        }
    }
}
```

Sessió 2:

Codi:

ExceptionMedicament.java

```
package Sessio2;

public class ExceptionMedicament extends Exception {
    public ExceptionMedicament(){super();}
    public ExceptionMedicament(String msg){super(msg);}
}
```

Malalt.java

```
package Sessio2;
import java.util.Arrays;

public class Malalt implements Comparable<Object> {
    private String nom;
    private MedicamentPindoles[] medicaments;
    private int num;
    private final int increment;

    public Malalt(String nom, int numMax, int increment) {
        if(nom==null || nom=="")     throw new
IllegalArgumentException("Argument nom illegal!");
        if(numMax<=0 )                        throw new
IllegalArgumentException("Argument numMax illegal! (ha de ser major a 0)");
        if(increment<=0)                throw new
IllegalArgumentException("Argument increment illegal! (ha de ser major a
0)");


        this.nom = nom;
        medicaments = new MedicamentPindoles[numMax];
        this.increment = increment;
    }
```

```
    public void comprarMedicamentPindoles(Object p) {
        if (!(p instanceof MedicamentPindoles))
              throw new IllegalArgumentException("Heu de passar un objecte
de MedicamentPindoles!");

        if(num==medicaments.length) ampliar();
        medicaments[num]=(MedicamentPindoles) p; num++;
        ordenar();
    }
    public void comprarMedicamentPindoles(String p, int pindoles) {
        // La comprovacio ja es fa en el constructor, innecesari ferla 2
vegades.
        MedicamentPindoles nouMedicament = new MedicamentPindoles(p,
pindoles);
        comprarMedicamentPindoles(nouMedicament);
    }

    public int totalPindolesQueden() {
        int n=0;
        for(int i=0; i<num; i++) n+=medicaments[i].quantesUnitatsQueden();
        return n;
    }
    public int totalPindolesPreses(String nom){
        if (nom==""||nom==null)
              throw new IllegalArgumentException("Nom incorrecte!");

        for(int i=0; i<num; i++) {
              if(medicaments[i].equals(nom)) return
medicaments[i].quantesUnitatsQueden();
        }
        throw new IllegalArgumentException("No s'ha torbat medicament amb el
nom \""+nom+"\" ");
    }

    public int maximPindoles(){
        int n=0;
        for(int i=0; i<num; i++) {
              if(medicaments[i].quantesUnitatsQueden()>n)
```

```
                    n=medicaments[i].quantesUnitatsQueden();
        }
        return n;
    }

    public MedicamentPindoles medicamentMenys(){
        int x=0;
        for(int i=1; i<num; i++) {
              int comp=medicaments[i].compareTo(medicaments[x]);
              if(comp<0 || (comp==0 &&
medicaments[i].getNom().compareTo(medicaments[x].getNom())<0) )
                    x=i;
        }
        return medicaments[x];
    }

    public String numMedicamentsPerQueden() {
        int queden[]=new int[num]; int quants[]=new int[num]; int quantes=0;

        for(int i=0; i<num; i++) {
              int x=troba(medicaments[i], queden, quantes); int posicio=x;

              if(x==-1 || quantes==0){
                    queden[quantes]=medicaments[i].quantesUnitatsQueden();
                    posicio=quantes; quantes++;
              }
              quants[posicio]++;
        }
        return crear(queden,quants,quantes);
    }

    public int eliminarMedicamentPindoles() {
        int quants=0; int eliminats=0;
        for(int i=0; i<num; i++) {
              if(medicaments[i].quantesUnitatsQueden()==0){
                    medicaments[i]=null; eliminats++;
              } else {
                    medicaments[quants]=medicaments[i]; quants++;
```

```
            }
        }
        return eliminats;
    }

    public MedicamentPindoles[] donaMedicamentsBuits() {
        MedicamentPindoles[] buits_tmp=new MedicamentPindoles[num]; int
cnt=0;
        for (MedicamentPindoles i : medicaments){
            if (i==null) continue; // legal? si no ho es, negar el if i
que envolti tot.
            if (i.quantesUnitatsQueden()==0){
                buits_tmp[cnt]=i; cnt++;
            }
        }

        MedicamentPindoles[] buits=new MedicamentPindoles[cnt];
        for (int x=0;x<cnt;x++) buits[x]=buits_tmp[x];
        return buits;
    }

    public void llistatOrdenatAscendent() {
        MedicamentPindoles[] aOrdenar=new MedicamentPindoles[num]; int
cnt=0;
        copiar(medicaments, aOrdenar, num);
        Bombolla(aOrdenar);
        for (MedicamentPindoles i: aOrdenar) {
            cnt++; System.out.println(cnt+". "+i);
        }
    }
    public void llistatOrdenatDescendent() {
        MedicamentPindoles[] aOrdenar=new MedicamentPindoles[num]; int
cnt=0;
        copiar(medicaments, aOrdenar, num);
        Arrays.sort(aOrdenar);
        for (Object i: reverse(aOrdenar)) {
            cnt++; System.out.println(cnt+". "+i);
        }
```

```
    }

    // OVERWRRIDEs
    public String toString() {
       String msg="";
       for (int x=0;x<num;x++) msg+=(x+1)+". "+medicaments[x]+"\n";
       return msg;
    }
    public boolean equals(Object malaltB){
       if (!(malaltB instanceof Malalt))
             return false;

       String[] llista_strMalaltA=numMedicamentsPerQueden().split("-");
       String[]
llista_strMalaltB=((Malalt)malaltB).numMedicamentsPerQueden().split("-");

       Arrays.sort(llista_strMalaltA); Arrays.sort(llista_strMalaltB);

       String strMalaltA=Arrays.toString(llista_strMalaltA);
       String strMalaltB=Arrays.toString(llista_strMalaltB);

       return strMalaltA.equals(strMalaltB);
    }
    public int compareTo(Object o) {
       if(!(o instanceof Malalt)) throw new ClassCastException("Tipus
incorrecte!");
       return this.totalPindolesQueden()-((Malalt)
o).totalPindolesQueden();
    }

    //GETs
    public String getNom() {return nom;}
    public int getNum() {return num;}
    public int getIncrement() {return increment;}
    public MedicamentPindoles getMedicamentPindoles(int quin){
       if(quin < num) return medicaments[quin];
       throw new IllegalArgumentException("No existeix medicament amb
l'index "+quin);
```

```
    }
    public MedicamentPindoles getMedicamentNoBuit() {
        for(int i=0; i<num; i++) {
              if(medicaments[i].quantesUnitatsQueden()>0) return
medicaments[i];
        }
        return null;
    }

    //PRIVATEs
    private void ampliar() {
        MedicamentPindoles[] nousMedicaments = new
MedicamentPindoles[medicaments.length + increment];
        for(int i = 0; i<this.medicaments.length; i++) nousMedicaments[i] =
medicaments[i];
        medicaments = nousMedicaments;
    }
    private void ordenar() {
        MedicamentPindoles aux;
        for(int x=0; x<num; x++){
              for(int y=num-1; y>x; y--){
                    int
comparacio=medicaments[y].getNom().compareTo(medicaments[y-1].getNom());

                    aux=medicaments[y];
                    int pindolesPreses1=0; int pindolesPreses2=0;
                    int pindolesMax1=0; int pindolesMax2=0;
                    if (comparacio==0) {

pindolesPreses1=medicaments[y].getPindolesPreses();

pindolesPreses2=medicaments[y-1].getPindolesPreses();


pindolesMax1=pindolesPreses1+medicaments[y].quantesUnitatsQueden();

pindolesMax2=pindolesPreses2+medicaments[y-1].quantesUnitatsQueden();
                    }
```

```
                    if (comparacio<0 || (comparacio==0 &&
(pindolesMax1>pindolesMax2 || (pindolesMax1==pindolesMax2 &&
pindolesPreses1>pindolesPreses2)))){
                        medicaments[y]=medicaments[y-1];
medicaments[y-1]=aux;
                    }
                }
            }
        }
    private void Bombolla(Comparable<Object> aOrdenar[]) {
        Comparable<Object> aux;
        for(int i=1; i<num-1; i++) {
              for(int j=num-1; j>=i; j--) {
                    if(aOrdenar[j].compareTo(aOrdenar[j-1]) < 0) {

                        aux = aOrdenar[j];
                        aOrdenar[j] = aOrdenar[j-1];
                        aOrdenar[j-1] = aux;
                    }
              }
        }
    }
    private void copiar(MedicamentPindoles[] origen, MedicamentPindoles[]
desti, int n) {
        for (int i=0; i<n; i++) {
              desti[i] = origen[i];
        }
    }
    private Object[] reverse(Object[] taula) {
        Object aux[] =new Object[taula.length];
        for (int i=0; i<taula.length; i++) aux[i]=taula[(taula.length-1)-i];
        return aux;
    }
    private static int troba(MedicamentPindoles caixa, int[] queden, int
quants) {
        int quantitat=caixa.quantesUnitatsQueden();
        for(int i=0; i<quants; i++) {
```

```
            if (queden[i]==quantitat) return i;
        }
        return -1;
    }
    private static String crear(int[] queden, int[] quants, int quantes) {
        String msg="";
        for(int i=0; i<quantes; i++)
            msg+="Amb "+queden[i]+" píndoles queden "+quants[i]+"
medicaments - ";

        return msg;
    }
}
```

MedicamentPindoles.java

```
package Sessio2;

public class MedicamentPindoles implements Comparable<Object> {
    private final String nom;
    private final int unitats;
    private int preses = 0;

    public MedicamentPindoles(String nom, int unitats){
        if(nom==null || nom=="") throw new
IllegalArgumentException("Argument nom illegal!");
        if(unitats<=0) throw new IllegalArgumentException("Argument unitats
illegal! (ha de ser major o igual a 0)");


this.nom=nom.substring(0,1).toUpperCase()+nom.substring(1).toLowerCase();
        this.unitats=unitats;
    }

    // GETs
    public int quantesUnitatsQueden() {return unitats-preses;}
    public void prendrePindola(){
        try{
```

```
            if(preses < unitats) preses++;
            else throw new ExceptionMedicament("No queden més
píndoles!");
        } catch (Exception e) {
            System.out.println(e);
        }
    }
    public int getPindolesPreses() {return preses;}
    public String getNom() {return nom;}

    // OVERWRRIDES
    public boolean equals(Object o) {
        // no cal fer 2 metodes, 1 per String i 1 per Object.
        if(o instanceof MedicamentPindoles) {
            return this.getNom().equals(((MedicamentPindoles)
o).getNom());
        } else if (o instanceof String) {
            return this.getNom().equals(o);
        } else {
            return false;
        }
    }
    public String toString() {
        return "Nom del medicament " + nom + " amb " +
quantesUnitatsQueden() + " pindoles de " + unitats;
    }
    public int compareTo(Object o){
        if(!(o instanceof MedicamentPindoles))
            throw new ClassCastException("Tipus incorrecte!");

        return this.quantesUnitatsQueden()-((MedicamentPindoles)
o).quantesUnitatsQueden();
    }
}
```

Prova.java

```
package Sessio2;
```

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

public class Prova {
	public static void main(String args[]){
	main2();
	}
	public static void main1() {
	 System.out.println("Errors:");

	 System.out.println("\nMalalts:");
	 try {new Malalt("",1,1);} catch (IllegalArgumentException e)
{System.out.println(e);}
	 try {new Malalt("Nom",0,1);} catch (IllegalArgumentException e)
{System.out.println(e);}
	 try {new Malalt("Nom",1,-1);} catch (IllegalArgumentException e)
{System.out.println(e);}



	 Malalt malalt=new Malalt("Jon", 10, 2);

	 System.out.println("\nMedicaments:");
	 try {malalt.comprarMedicamentPindoles("Error :)");} catch
(IllegalArgumentException e) {System.out.println(e);}
	 try {malalt.comprarMedicamentPindoles(null, 3);} catch
(IllegalArgumentException e) {System.out.println(e);}
	 try {malalt.comprarMedicamentPindoles("NomMedicament", -1);} catch
(IllegalArgumentException e) {System.out.println(e);}

	 malalt.comprarMedicamentPindoles("Ibuprufeno", 20);
	 malalt.comprarMedicamentPindoles("Paracetamol", 13);

	 MedicamentPindoles med=malalt.getMedicamentNoBuit();
	 for (int x=(int)med.quantesUnitatsQueden()/2; x>0; x--)
		med.prendrePindola();
```

```
        System.out.println("\ntotalPindolesPreses:");
        try { malalt.totalPindolesPreses(null); } catch
(IllegalArgumentException e) {System.out.println(e);}
        try { malalt.totalPindolesPreses("Nom inexistent"); } catch
(IllegalArgumentException e) {System.out.println(e);}


        System.out.println("\ncompareTo:");
        try { malalt.compareTo((Integer) 1); } catch (ClassCastException e)
{System.out.println(e);}


        System.out.println("\ngetMedicamentPindoles:");
        try { malalt.getMedicamentPindoles(3); } catch
(IllegalArgumentException e) {System.out.println(e);}
      }


      public static final String[] nomsMalalts={
        "Josep",
        "Josepa",
        "Joseph",
        "Josephina"
    };
      public static final String[] nomsMedicaments={
        "Ibuprufeno",
        "Paracetamol",
        "Aspirina",
        "Omeprazol",
        "Amlodipina"
    };

      public static void main2() {
        System.out.print("Quants malalts: "); int
numMalalts=Keyboard.readInt();
        Malalt[] malalts=new Malalt[numMalalts]; int cntMalalts=0;
```

```
        for(int i=0;i<numMalalts;i++){
              Random rand=new Random();

              String
nomMalalt=nomsMalalts[rand.nextInt(nomsMalalts.length)];
              int maximMedicaments=rand.nextInt(10)+1;
              int numMedicaments=rand.nextInt(maximMedicaments)+1;
              int increment=rand.nextInt(5)+1;

              Malalt malalt=new Malalt(nomMalalt, maximMedicaments,
increment);

              for (int o=0;o<numMedicaments; o++)

malalt.comprarMedicamentPindoles(nomsMedicaments[rand.nextInt(nomsMedicamen
ts.length)], rand.nextInt(50)+1);

              print(malalt);

              malalts[cntMalalts]=malalt; cntMalalts++;
        }


System.out.println("\n\n\n\n**********************************************
******************");

System.out.println("**************************************************
***********\n\n");

        Arrays.sort(malalts);
        for(Malalt m : malalts) print(m);

       }

       public static void print(Malalt malalt) {
        System.out.println("\n\n*************** "+malalt.getNom()+"
***************");
        System.out.println("Malalt "+malalt.getNom()+": ");
```

```
System.out.println(malalt);
        System.out.println("Llistat Ordenat Descendent: ");
malalt.llistatOrdenatDescendent();
        System.out.println("\nLlistat Ordenat Ascendent: ");
malalt.llistatOrdenatAscendent();
        // System.out.println("\nnumMedicamentsPerQueden:\n"+
malalt.numMedicamentsPerQueden());
       }
}
```

Organització de la feina:

En aquesta pràctica n'hem dividit les diferents classes en cada sessió per tal d'anar fent-ho paral·lelament amb l'objectiu d'acabar més de pressa, una vegada acabàvem les nostres classes, les posàvem en comú per tal d'entendre que havia fet l'altre i detectar possibles errors o afegir possibles optimitzacions.

Valoració:

Pel que fa a la valoració del funcionament dels exercicis desenvolupats en aquesta primera pràctica, val a dir que no hem tingut cap problema greu per a resoldre'ls.