Corrigé des exos de TD- Série n°3

Exercice 2

1/ class Point **{**double x, y ;
Point (double a, double b) { x= a ; y = b ; }
void **Deplacer** (double a, double b) { x= x+a ; y = y+b ;}
double **Distance** (Point p)
{ double d= (p.x – x)*(p.x - x) + (p.y – y)*(p.y – y) ; return (Math.sqrt(d)); }
void **Afficher()**

{ System.out.println (″(″+ x + ″,″ + y + ″)″) ;}

}

2/ class ObjGeog **{**Point localisation ; String forme ; String couleur ;
ObjGeog (Point loc, String f, String c)
{ localisation = new Point (loc.x, loc.y); forme = f ; couleur = c ;}
void **afficher** ()
{ System.out.print ("les caractéristiques de l'objet geog sont : localisation =") ;
localisation.Afficher() ; System.out.println (" forme =" + forme + " couleur =" + couleur) ;
}
}

3/ enum Etat { bon, moyen, dégradé } ;
class Route extends ObjGeog
{int Longueur; **Etat** etat ;
Route (Point loc, String f, String c, int lg, String e)
{ **super (loc, f, c) ;**
longueur = lg ; etat = **Etat**.valueOf (e); }
}

---

class Batiment extends ObjGeog
{ String categorie ; int annee;
Batiment (Point loc, String f, String c, String cat, int an)
{ **super (loc, f, c) ;**
categorie = cat ; annee = an ;}
void **afficher** ()
{super. Afficher() ;
System.out.println ("les caractéristiques du batiment sont : catégorie=" +
categorie + "année de construction =" + annee) ; } **}**

4/ On utilise un objet de la classe **Vector** ou un objet de la classe **ArrayList** (*voir les deux classes dans le chapitre 6*)

import java.util.Vector ; import java.util.Scanner;

class ProgramObjGeog

```
 { public static void main(String arg[])
      { Vector <Batiment>   B = new  Vector <Batiment> () ;
        Scanner e = new  Scanner (System.in);    Batiment   b;
String  f, c, cat; int annee;  Point  loc; double  x, y; System.out.println ("donnez l'année  A
pour l'arrêt de la saisie :") ; int A = e.nextInt() ;

while(1)
        {  System.out.println ("donnez la description d'un bâtiment ") ;
           X = e.nextDouble() ;  y = e.nextDouble() ;  loc = new  Point(x, y) ;
           f= e.next() ;  c= e.next() ; cat = e.next() ; annee = e.nextInt() ;
           if (annee = A)   break;
                 else  {   // création et stockage d'objet
                           b = new  Batiment (loc, f, c, cat, annee);    B.add(b); }}
```

```
5/    // batiments  se trouvant à une distance  <d  de la route R
System.out.println ("donnez la distance d ") ;   double d = e.nextDouble() ;
Point  p = new Point (2, 10) ;    Point  p1;
for  (i = 0 ; i< B.size() ; i ++)
  { p1 = B.elementAt(i).localisation;
    if  (p1.Distance (p) <d)      B.elementAt(i). Afficher(); }  }
    }
```

---

*// la même solution en utilisant un objet de la classe **ArrayList***

```
Import java.util.ArrayList;
class ProgramObjGeog
 { public static void main(String arg[])
      { ArrayList  <Batiment>   L = new  ArrayList <Batiment> () ;

          System.out.println ("donnez l'année  A ") ;  int A = e.nextInt() ;
       While(1)
            {  System.out.println ("donnez la description d'un bâtiment ") ;
            X = e.nextDouble() ;  y = e.nextDouble() ;     loc = new  Point(x, y) ;
           f= e.next() ;  c = e.next() ; cat = e.next() ; annee = e.nextInt() ;
            if (annee = = A)   break;
                 else  {  b = new  Batiment (loc, f, c, cat, annee);    L.add(b); }  }

5/    // batiments  se trouvant à une distance  <d  de la route R
System.out.println ("donnez la distance d ") ;   double d = e.nextDouble() ;
Point  p = new Point (2, 10) ;    Point  p1;
for  (i = 0 ; i< L.size() ; i ++)
  { p1 = L.get(i).localisation;     if  (p1.Distance (p) <d)     L.get(i). Afficher(); }   } }
```

Exercice 4

```
1/  class Date     { int Jour, Mois, Annee;
                                  // constructeur
                    Date  ( int J, int M, int A)        { Jour = J ;  Mois = M ; Annee = A ; }
```

```
boolean Inferieure ( Date d)
      { if (this.Annee < d.Annee)    return (true);
            else if (this.Annee > d.Annee)   return (false);
                                 else  // meme année
                                   if   (this.Mois < d.Mois)    return (true);
                                         else if (this.Mois > d.Mois)    return (false);
                                        else  // meme mois
                                         if (this.Jour < d.Jour)     return (true);
                                           else            return (false) ; }

void Afficher ()   { System.out.println (Jour + " /"  + Mois + " /"  Annee) ;}
 } // fin de la classe Date
```

```
1/  class Heure { int heure, minute, seconde;
                                    // constructeur
                  Heure  ( int h, int m, int s)        { heure= h ;  minute = m ; seconde = s ; } }

void Afficher()   { System.out.println (heure + " : " + minute + " : "  seconde) ;}
 } // fin de la classe Heure
```

```
3/ class Message
{ int numero, String objet, contenu;     private String  expediteur, destinataire ;
private Date Dat; Heure H ;  private float taille ;
                                                  // constructeur
 Message  ( int num, , String obj, String exp, String dest, Date d, Heure h, float t)
{ numero = num ; objet = obj ; expediteur = exp ; destinataire = dest ;
  Dat = new Date (Dat.jour, Dat.Mois, Dat.Annee) ;
  Heure = new Heure (H.heure, H.minute, H.seconde) ;}
                                                    // Accessueurs
   void  setTaille (float t)   {taille = t;}
    void setDate (Date d) {Dat.Jour = d.jour ; Dat.mois = d.mois ; Dat.annee = d.annee}
   void  setExpediteur (String  exp)    { Expediteur = exp;}
   void  setDestinataire (String  exp)    { Destinataire= dest;}    ...
   float  getTaille()   {return (taille);}

    Date getDate () {return (Dat) ; }
   String getExpediteur ()    { return (Expediteur ) ;}
   String getDestinataire ()  { return (Destinataire);}

void Afficher ()   { System.out.println (numero + "    " +  objet  + "   "  expediteur + "   "  destinataire+
"   "  taille) ;  Dat. Afficher() ; H.Afficher() ;}
void AfficherContenu  ()   { System.out.println (contenu) ;}
 } // fin de la classe Message
```

```
4/ class MessageRecu extends Message
{ private boolean Etat ;
                                                  // constructeur
 MessageRecu ( int num, , String obj, String exp, String dest, Date d, Heure h, float t)
   {  super (num, obj, exp, dest, d, h, t);   Etat = false;}
                                                    // Accessueurs
   void  setEtat ()   {Etat = true;}
   boolean getEtat () {return (Etat) ;}

void Afficher ()   { super.Afficher() ;  System.out.println (Etat) ;}
 } // fin de la classe MessageRecu
```

```
5/ class Boite_Messages
{  String adresse, int nbRecu, nbEnvoi, nbNonLu ; float Espace ;
 Vector < MessageRecu> MessRecu = new Vector < MessageRecu> () ;
  Vector < Message> MessEnv = new Vector < Message> () ;
                                                    // constructeur
 Boite_Message  (String adr, int nb1, int nb2, int nb3, float e)
{ adr = adresse ; nbRecu = nb1 ; nbEnvoi= nb2 ; nbNonLu = nb3 ; Espace = e ;}

void EnvoyerMessage (Message m)
  { MessEnv.add (m) ; nbEnvoi++ ;  Espace = Espace – m.getTaille() ; }
// ajouter le message au vecteur MessEnv, mettre à jour le nbre et l'espace de stockage restant

void RecevoirMessage (Message m)
  { MessRecu.add (m) ; nbRecu++ ;  nbNonLu++ ; Espace = Espace – m.getTaille() ; }
/* ajouter le message au vecteur MessRecu, mettre à jour nbRecu et nbNonLu et l'espace de stockage restant */

void LireMessage (int num)
  { for (int i= 0 ; i< MessRecu.size(); i++) { if (MessRecu.elementAt(i).numero = = num)
                                                  MessRecu.elementAt(i).AfficherContenu(); nbNonLu --}
// parcourir le  vecteur MessRecu pour chercher le numéro de message et afficher son contenu

void AfficherRecus ()
  { for (int i= 0 ; i< MessRecu.size(); i++)  MessRecu.elementAt(i).Afficher();}

void AfficherEnvoyes ()
  { for (int i= 0 ; i< MessEnv.size(); i++)  MessEnv.elementAt(i).Afficher();}
  } // fin de la classe Boite_Message
```

```
6/  import  java.util.Vector ; import   java.util.Scanner;
class ProgramBoiteMessage
  { public static void main(String arg[])
        { Scanner e = new  Scanner (System.in);    String Adr = e.next() ;
          Boite_Messages B = new Boite_Messages (Adr, 0, 0, 0, 1024) ;
          for  (i = 0 ; i< 100 ; i++)
               {// lecture de données pour un message reçu
                 MessageRecu m = new MessageRecu (num, obj, exp, dest , d, h, t) ;
                 B.RecevoirMessage(m); }
           for  (i = 0 ; i< 20 ; i++)
               {// lecture de données pour un message envoyé
                 Message m = new Message (num, obj, exp, dest, d, h, t) ;
                 B.EnvoyerMessage(m); }
// lire un message donné

int num = e.nextInt() ;     B.LireMessage(num);

// afficher les messages reçus après une date d
int j = e.nextInt() ;    int m = e.nextInt() ;    int a = e.nextInt() ;
Date d = new Date (j, m, a);
for (i = 0; i<MessRecu.size(); i++)
       if (d. Inferieure (MessRecu.elementAt(i).getDate()))
                MessRecu.elementAt(i). Afficher() ;}}
```