Algorithmique & Programmation
Semestre 2 ST
Les types agrégés

Rappels du semestre 1

Types agrégés	Structuration en sous-algorithmes	Tableaux de variables	Algorithmes de recherche et de tri	Précision, rapidité et complexité
Matrices de variables	Récursivité	Travaux dirigés	Travaux pratiques	Informations
Sujets de projet	Evaluation intermédiaire	Evaluation finale	Evaluation complémentaire	Archives

Cours TD TP - Corrections

Version PDF

Clavier.class - Ecran.class - Documentation

Partie 1 : Implantation des exercices de TD

Tous les exercices sont à réaliser en langage Java.

Exercice n°1

Pour une fédération sportive, on souhaite développer un programme de gestion des licences.
Concevoir un type agrégé "licence" de stockage des informations relatives à un licencié:

- nom,
- prénom,
- numéro de licence,
- club,
- vétéran ou non.

TypeLicence.java

```
/* Type agrege de stockage des informations     */
/* relatives a un licencie sportif              */

static class Licence {
   String nom = "";
   String prenom = "";
   int numeroLicence = 0;
   String club = "";
   char categorieAge = ' ';
   boolean sexe = true; };
```

Exercice n°2

a) Concevoir un type agrégé "position3D" de stockage d'une position définie dans un espace à trois dimensions réelles.

b) Concevoir un algorithme de lecture au clavier d'une variable de type position3D, puis d'affichage à l'écran de cette variable.

LectureEcriturePosition3D.java

```
/* Lecture et ecriture d'une Position3D         */

public class LectureEcriturePosition3D {

/* Type agrege de stockage des informations     */
/* relatives a une position en trois dimensions */

   static class Position3D {
     double x = 0.0;
     double y = 0.0;
     double z = 0.0; };

/* Programme principal                          */

   public static void main(String [] args) {
     Position3D pos = new Position3D();
    Ecran.afficherln("SVP, vos 3 coordonnees?");
    pos.x = Clavier.saisirDouble();
    pos.y = Clavier.saisirDouble();
    pos.z = Clavier.saisirDouble();
    Ecran.afficher("(",pos.x,",",pos.y,",",pos.z,")");
  }
}
```

Exemple d'exécution

Exercice n°3

a) Concevoir un type agrégé "temps" de stockage d'une heure représentée par un nombre d'heures, de minutes et de secondes.

b) Concevoir un algorithme réalisant les traitements suivants:
- Lecture au clavier de deux variables de type temps.
- Affichage de l'heure la plus tardive parmis les deux, puis affichage de l'autre heure.
- Affichage du nombre de secondes entre les deux.

ManipulationTemps.java

```
/* Manipulations sur une classe temps          */

public class ManipulationTemps {

/* Type agrege de stockage des informations    */
/* relatives a un temps code sous la forme     */
/* heure, minute, seconde                      */

   static class Temps {
     int h = 0;
     int mn = 0;
     int s = 0; };

/* Programme principal                         */

   public static void main(String [] args) {
     Temps t1 = new Temps();
     Temps t2 = new Temps();
     boolean inferieurEgalT1T2;
     int difference;
     int s1;
     int s2;
    Ecran.afficherln("SVP, t1 (h, mn & s)?");
    t1.h = Clavier.saisirInt();
    t1.mn = Clavier.saisirInt();
    t1.s = Clavier.saisirInt();
    Ecran.afficherln("SVP, t2 (h, mn & s)?");
    t2.h = Clavier.saisirInt();
    t2.mn = Clavier.saisirInt();
    t2.s = Clavier.saisirInt();
    s1 = 3600*t1.h+60*t1.mn+t1.s;
    s2 = 3600*t2.h+60*t2.mn+t2.s;
    inferieurEgalT1T2 = (s1 <= s2);
     if ( inferieurEgalT1T2 ) {
      difference = s2-s1;
      Ecran.afficherln("[",t2.h,":",t2.mn,":",t2.s,"]");
      Ecran.afficherln("[",t1.h,":",t1.mn,":",t1.s,"]"); }
```

```
        else {
       difference = s1-s2;
       Ecran.afficherln("[",t1.h,":",t1.mn,":",t1.s,"]");
       Ecran.afficherln("[",t2.h,":",t2.mn,":",t2.s,"]"); }
     Ecran.afficherln("Ecart : ",s2-s1," secondes");
   }
}
```

Exemple d'exécution

Exercice n°4

a) Concevoir un type agrégé "couleur" de stockage d'une couleur représentée par une valeur de rouge, une valeur de vert et une valeur de bleu. Ces trois valeurs sont de type entier. Usuellement en informatique, elles sont comprises entre 0 et 255 (8 bits).
Exemples: (255,255,255) -> blanc, (0,0,0) -> noir, (255,0,0) -> rouge, (255,255,0) -> jaune

b) Concevoir un algorithme réalisant les traitements suivants:
- Lecture au clavier d'une variable de type couleur.
- Lecture au clavier d'un pourcentage d'assombrissement.
- Calcul et affichage de cette couleur après assombrissement de la valeur lue.
- Lecture au clavier d'un pourcentage d'éclaircissement.
- Calcul et affichage de cette couleur après éclaircissement de la valeur lue.

ManipulationCouleurs.java

```
/* Manipulations sur une classe couleur         */

public class ManipulationCouleurs {

/* Type agrege de stockage des informations     */
/* relatives a une couleur codee sous la forme */
/* heure, minute, seconde                       */

   static class Couleur {
     short r = 0;
     short v = 0;
     short b = 0; };

/* Programme principal                          */

   public static void main(String [] args) {
     Couleur c = new Couleur();
     double assombrissement;
     Couleur ca = new Couleur();
```

```
      double eclaircissement;
      Couleur ce = new Couleur();
      double fa;
      double fe;
     Ecran.afficherln("SVP, r, v & b?");
     c.r = Clavier.saisirShort();
     c.v = Clavier.saisirShort();
     c.b = Clavier.saisirShort();
     Ecran.afficherln("SVP, assombrissement?");
     assombrissement = Clavier.saisirDouble();
     Ecran.afficherln("SVP, eclaircissement?");
     eclaircissement = Clavier.saisirDouble();
     fa = 1.0-assombrissement/100.0;
     ca.r =(short) Math.round(fa*c.r);
     ca.v =(short) Math.round(fa*c.v);
     ca.b =(short) Math.round(fa*c.b);
     fe = 1.0-eclaircissement/100.0;
     ce.r =(short) (255-Math.round(fe*(255-c.r)));
     ce.v =(short) (255-Math.round(fe*(255-c.v)));
     ce.b =(short) (255-Math.round(fe*(255-c.b)));
     Ecran.afficherln("[",c.r,":",c.v,":",c.b,"]");
     Ecran.afficherln("[",ca.r,":",ca.v,":",ca.b,"]");
     Ecran.afficherln("[",ce.r,":",ce.v,":",ce.b,"]");
   }
}
```

Exemple d'exécution

Exercice n°5

a) Concevoir un type agrégé "position2D" de stockage d'une position définie dans un espace à deux dimensions réelles.

b) Concevoir un type agrégé "triangle2D" de stockage d'un triangle de 3 position2D.

c) Concevoir un algorithme réalisant les traitements suivants:
- Lecture au clavier d'un triangle2D.
- Calcul et affichage de la surface de ce triangle2D.

ManipulationTriangle2D.java

```
/* Manipulations sur une classe triangle       */
/* en deux dimensions                          */

public class ManipulationTriangle2D {

/* Type agrege de stockage des informations    */
/* relatives a une position en deux dimensions */
```

```
   static class Position2D {
     double x = 0.0;
     double y = 0.0; };

/* Type agrege de stockage des informations    */
/* relatives a un triangle en deux dimensions  */

   static class Triangle2D {
     Position2D p0 = new Position2D();
     Position2D p1 = new Position2D();
     Position2D p2 = new Position2D(); };

/* Programme principal                         */

   public static void main(String [] args) {
     Triangle2D t = new Triangle2D();
     double surface;
    Ecran.afficherln("SVP, x & y du sommet 1?");
    t.p0.x = Clavier.saisirDouble();
    t.p0.y = Clavier.saisirDouble();
    Ecran.afficherln("SVP, x & y du sommet 2?");
    t.p1.x = Clavier.saisirDouble();
    t.p1.y = Clavier.saisirDouble();
    Ecran.afficherln("SVP, x & y du sommet 3?");
    t.p2.x = Clavier.saisirDouble();
    t.p2.y = Clavier.saisirDouble();
    surface = (t.p0.x-t.p1.x)*(t.p0.y+t.p1.y)+
              (t.p1.x-t.p2.x)*(t.p1.y+t.p2.y)+
              (t.p2.x-t.p0.x)*(t.p2.y+t.p0.y);
    surface = Math.abs(surface)/2.0;
    Ecran.afficherln("Surface: ",surface);
  }
}
```

Exemple d'exécution

Exercice n°6

a) Concevoir un type agrégé "parallelepipede" de stockage des caractéristiques d'un parallélépipède rectangle à faces perpendiculaires aux axes.

b) Concevoir un algorithme réalisant les traitements suivants:
- Lecture d'un parallelepipede.
- Calcul et affichage du volume de ce parallelepipede.

Le volume d'un parallelepipede est égal au produit de la longueur de ses cotés.

ManipulationCube.java

```
/* Manipulations sur une classe parallelepipede */
/* rectangle a faces orthogonales aux axes      */

public class ManipulationCube {

/* Type agrege de stockage des informations     */
/* relatives a une position en trois dimensions */

   static class Position3D {
     double x = 0.0;
     double y = 0.0;
     double z = 0.0; };

/* Type agrege de stockage des informations     */
/* relatives a un parallelepipede rectangle     */
/* a faces orthogonales aux axes                */

   static class Parallelepipede {
     Position3D min = new Position3D();
     Position3D max = new Position3D(); }

/* Programme principal                          */

   public static void main(String [] args) {
     Parallelepipede cb = new Parallelepipede();
     double volume;
    Ecran.afficherln("SVP, x, y & z du sommet min?");
    cb.min.x = Clavier.saisirDouble();
    cb.min.y = Clavier.saisirDouble();
    cb.min.z = Clavier.saisirDouble();
    Ecran.afficherln("SVP, x, y & z du sommet max?");
    cb.max.x = Clavier.saisirDouble();
    cb.max.y = Clavier.saisirDouble();
    cb.max.z = Clavier.saisirDouble();
    volume = (cb.max.x-cb.min.x)*
             (cb.max.y-cb.min.y)*
             (cb.max.z-cb.min.z);
    Ecran.afficherln("Volume: ",volume);
  }
}
```

Exemple d'exécution

Exercice n°7

a) Concevoir un type agrégé "date" de stockage d'une date représentée par un numéro de jour, un numéro de mois et un numéro d'année.

b) Concevoir un algorithme réalisant les traitements suivants:
- Lecture d'une date.
- Calcul de la date du lendemain.
- Affichage de cette date.

ManipulationDates.java

```
/* Manipulations sur une classe date          */

public class ManipulationDates {

/* Type agrege de stockage des informations   */
/* relatives a une date codee sous la forme   */
/* jour, mois, annee                          */

   static class Date {
     int j = 1;
     int m = 1;
     int a = 1901; };

/* Programme principal                        */

   public static void main(String [] args) {
     Date dt = new Date();
     int nbJours;
    Ecran.afficherln("SVP, j, m & a?");
    dt.j = Clavier.saisirInt();
    dt.m = Clavier.saisirInt();
    dt.a = Clavier.saisirInt();
     switch (dt.m) {
       case 2 :
         if ( ( (dt.a%400) == 0 ) || ( ( (dt.a%
4) == 0 ) && ( (dt.a%100) != 0 ) ) ) {
          nbJours = 29; }
           else {
          nbJours = 28; }
        break;
       case 4  :
       case 6  :
       case 9  :
       case 11 :
        nbJours = 30;
         break;
       default :
        nbJours = 31;
         break; }
    dt.j = dt.j+1;
     if ( dt.j > nbJours ) {
      dt.j = 1;
```

```
      dt.m = dt.m+1;
       if ( dt.m > 12 ) {
        dt.m = 1;
        dt.a = dt.a+1; } }
    Ecran.afficherln(dt.j,"/",dt.m,"/",dt.a);
  }
}
```

Exemple d'exécution

Exercice n°8

a) Concevoir un type agrégé quadrilatère (polygone à 4 sommets) en trois dimensions.

b) Concevoir un algorithme réalisant les traitements suivants:
- Lecture au clavier d'un quadrilatère
- Calcul puis affichage du périmètre de ce quadrilatère

ManipulationQuadrilatere.java

```
/* Manipulations sur une classe quadrilatere  */

public class ManipulationQuadrilatere {

/* Type agrege de stockage des informations   */
/* relatives a une position en 3 dimensions   */

   static class Position3D {
     double x = 0.0;
     double y = 0.0;
     double z = 0.0; };

/* Type agrege de stockage des informations   */
/* relatives a un quadrilatere en 3D          */

   static class Quadrilatere3D {
     Position3D p1 = new Position3D();
     Position3D p2 = new Position3D();
     Position3D p3 = new Position3D();
     Position3D p4 = new Position3D(); };

/* Programme principal                        */

   public static void main(String [] args) {
     double perimetre = 0.0;
     Quadrilatere3D q = new Quadrilatere3D();
    Ecran.afficherln
("SVP, les coordonnees de votre quadrilatere");
```

```
    Ecran.afficherln("Sommet 1");
    q.p1.x = Clavier.saisirDouble();
    q.p1.y = Clavier.saisirDouble();
    q.p1.z = Clavier.saisirDouble();
    Ecran.afficherln("Sommet 2");
    q.p2.x = Clavier.saisirDouble();
    q.p2.y = Clavier.saisirDouble();
    q.p2.z = Clavier.saisirDouble();
    Ecran.afficherln("Sommet 3");
    q.p3.x = Clavier.saisirDouble();
    q.p3.y = Clavier.saisirDouble();
    q.p3.z = Clavier.saisirDouble();
    Ecran.afficherln("Sommet 4");
    q.p4.x = Clavier.saisirDouble();
    q.p4.y = Clavier.saisirDouble();
    q.p4.z = Clavier.saisirDouble();
    perimetre = perimetre + Math.sqrt(Math.pow(q.p1.x-q.p2.x,2.0)
+
                                      Math.pow(q.p1.y-q.p2.y,2.0)
+
                                      Math.pow
(q.p1.z-q.p2.z,2.0));
    perimetre = perimetre + Math.sqrt(Math.pow(q.p2.x-q.p3.x,2.0)
+
                                      Math.pow(q.p2.y-q.p3.y,2.0)
+
                                      Math.pow
(q.p2.z-q.p3.z,2.0));
    perimetre = perimetre + Math.sqrt(Math.pow(q.p3.x-q.p4.x,2.0)
+
                                      Math.pow(q.p3.y-q.p4.y,2.0)
+
                                      Math.pow
(q.p3.z-q.p4.z,2.0));
    perimetre = perimetre + Math.sqrt(Math.pow(q.p4.x-q.p1.x,2.0)
+
                                      Math.pow(q.p4.y-q.p1.y,2.0)
+
                                      Math.pow
(q.p4.z-q.p1.z,2.0));
    Ecran.afficherln("Perimetre : ",perimetre);
  }
}
```

Exemple d'exécution

Exercice n°9

a) Concevoir un type agrégé "temps" de stockage d'une heure représentée par un nombre d'heures, de minutes et de secondes.

b) Concevoir un algorithme réalisant les traitements suivants:
- Lecture au clavier d'un temps t
- Lecture au clavier d'un nombre de secondes nbs
- Modification du temps t par incrémentation de nbs secondes
- Affichage écran de t

IncrementationTemps.java

```
/* Incrementation sur une classe temps          */

public class IncrementationTemps {

/* Type agrege de stockage des informations     */
/* relatives a un temps code sous la forme      */
/* heure, minute, seconde                       */

   static class Temps {
     int h = 0;
     int mn = 0;
     int s = 0; };

/* Programme principal                          */

   public static void main(String [] args) {
     Temps t = new Temps();
     int nbs;
    Ecran.afficherln("SVP, t (h, mn & s)?");
    t.h = Clavier.saisirInt();
    t.mn = Clavier.saisirInt();
    t.s = Clavier.saisirInt();
    Ecran.afficherln("SVP, Increment?");
    nbs = Clavier.saisirInt();
    t.s = t.s+nbs;
     if ( t.s >= 60 ) {
      t.mn = t.mn+(t.s/60);
      t.s = t.s%60;
       if ( t.mn >= 60 ) {
        t.h = (t.h+(t.mn/60))%24;
        t.mn = t.mn%60; } }
    Ecran.afficherln(t.h,":",t.mn,":",t.s);
  }
}
```

Exemple d'exécution

Exercice n°10

a) Concevoir un type agrégé "localisation géographique" avec nombres de degrés en entier, de minutes en entier et de secondes en réel.
Exemple: Horloge astronomique de la cathédrale de Besançon: Longitude 6°01'49.08", latitude 47°14'00.96"

b) Concevoir un algorithme permettant de convertir une localisation géographique en un nombre de degrés en réel (lecture au clavier de la localisation géographique, conversion et affichage du résultat de conversion). Concevoir un algorithme permettant de convertir un nombre de degrés en réel en une localisation géographique (lecture au clavier du nombre de degrés réel, conversion et affichage de la localisation géographique).

c) Concevoir un type agrégé "localisation GPS" avec longitude et latitude.

d) Concevoir un algorithme de calcul de la distance "à vol d'oiseau" existant entre deux localisations GPS définies à l'altitude 0.0.
Le rayon terrestre est de 6378.137 km. La formule de calcul est la suivante:

$dlo = (lo2 - lo1) / 2.0;$
$dla = (la2 - la1) / 2.0;$
$a = sin(dla)^2 + cos(la1)*cos(la2)*sin(dlo)^2;$
$d = 2.0 * atan2(sqrt(a), sqrt(1.0 - a)) * rayonTerrestre;$

Où lo1, lo2, la1 et la2 sont les longitudes et latitudes en réel des localisations GPS.

ManipulationCoordonneesGPS.java

```
/* Manipulations sur une classe                */
/* coordonnees GPS                             */

public class ManipulationCoordonneesGPS {

/* Type agrege de stockage des informations    */
/* relatives a une localisation codee          */
/* sous la forme degres, minutes, secondes     */

   static class Localisation {
     int d = 0;
     int mn = 0;
     double s = 0.0; };

/* Type agrege de stockage des informations    */
/* relatives a une localisation GPS            */

   static class LocalisationGPS {
```

```
      Localisation lon = new Localisation();
      Localisation lat = new Localisation(); };

/* Programme principal                          */

   public static void main(String [] args) {
      LocalisationGPS l1 = new LocalisationGPS();
      LocalisationGPS l2 = new LocalisationGPS();
      //l1.lon.d = -90;
      //l1.lon.mn = 30;
      //l1.lon.s = 10.0;
      //l1.lat.d = 0;
      //l1.lat.mn = 0;
      //l1.lat.s = 0.0;
      //l2.lon.d = 90;
      //l2.lon.mn = 30;
      //l2.lon.s = 10.0;
      //l2.lat.d = 0;
      //l2.lat.mn = 0;
      //l2.lat.s = 0.0;
     Ecran.afficherln("SVP, P1?");
     Ecran.afficherln("Longitude :");
     l1.lon.d = Clavier.saisirInt();
     l1.lon.mn = Clavier.saisirInt();
     l1.lon.s = Clavier.saisirDouble();
     Ecran.afficherln("Latitude :");
     l1.lat.d = Clavier.saisirInt();
     l1.lat.mn = Clavier.saisirInt();
     l1.lat.s = Clavier.saisirDouble();
     Ecran.afficherln("SVP, P2?");
     Ecran.afficherln("Longitude :");
     l2.lon.d = Clavier.saisirInt();
     l2.lon.mn = Clavier.saisirInt();
     l2.lon.s = Clavier.saisirDouble();
     Ecran.afficherln("Latitude :");
     l2.lat.d = Clavier.saisirInt();
     l2.lat.mn = Clavier.saisirInt();
     l2.lat.s = Clavier.saisirDouble();
      double lo1 = l1.lon.d+((l1.lon.s/60.0)+l1.lon.mn)/60.0;
      double lo2 = l2.lon.d+((l2.lon.s/60.0)+l2.lon.mn)/60.0;
      double la1 = l1.lat.d+((l1.lat.s/60.0)+l1.lat.mn)/60.0;
      double la2 = l2.lat.d+((l2.lat.s/60.0)+l2.lat.mn)/60.0;
     lo1 = lo1*Math.PI/180.0;
     lo2 = lo2*Math.PI/180.0;
     la1 = la1*Math.PI/180.0;
     la2 = la2*Math.PI/180.0;
      double dlo = (lo2-lo1)/2.0;
      double dla = (la2-la1)/2.0;
      double a = (Math.sin(dla)*Math.sin(dla))+Math.cos(la1)
*Math.cos(la2)*(Math.sin(dlo)*Math.sin(dlo));
```

```
      double distance = 2.0 * Math.atan2(Math.sqrt(a),Math.sqrt
(1.0-a)) * 6378.137;
    Ecran.afficherln(lo1);
    Ecran.afficherln(lo2);
    Ecran.afficherln(la1);
    Ecran.afficherln(la2);
    Ecran.afficherln("Distance : ",distance);
  }
}
```

Exemple d'exécution

Partie 2: Exercices suplémentaires

Exercice supplémentaire n°1

Une sphère est caractérisée par la position en 3D de son centre et son rayon.

a) Concevoir un type agrégé Java "Sphere" permettant de stocker une sphère.

b) Concevoir un algorithme réalisant les traitements suivants:
- Lecture de deux sphères.
- Calcul de la distance entre ces deux sphères (Distance minimale entre leurs surfaces).
- Affichage de cette distance.

Une droite dans un espace 3D peut être représentée par le système d'équations paramétriques la définissant.

c) Concevoir un type agrégé Java "Droite3D" permettant de stocker une droite 3D.

d) Concevoir un algorithme réalisant les traitements suivants:
- Lecture d'une sphère et d'une droite 3D.
- Test de l'existence d'une intersection entre cette sphère et cette droite.
- Affichage du résultat de ce test.

DistanceSpheres.java

```
/* Manipulations sur des spheres et des droites 3D */

public class DistanceSpheres {

/* Type agrege de stockage d'une position 3D       */

   static class Position3D {
     double x = 0.0;
```

```
      double y = 0.0;
      double z = 0.0; };

/* Type agrege de stockage d'une sphere 3D         */

   static class Sphere {
      double rayon = 1.0;
      Position3D centre = new Position3D(); };

/* Programme principal                              */

   public static void main(String [] args) {
      Sphere sp1 = new Sphere();
      Sphere sp2 = new Sphere();
      double dx;
      double dy;
      double dz;
      double distance;
     Ecran.afficherln("Premiere sphere");
     Ecran.afficher("Rayon                    ? ");
     sp1.rayon = Clavier.saisirDouble();
     Ecran.afficher("Coordonnee x du centre ? ");
     sp1.centre.x = Clavier.saisirDouble();
     Ecran.afficher("Coordonnee y du centre ? ");
     sp1.centre.y = Clavier.saisirDouble();
     Ecran.afficher("Coordonnee z du centre ? ");
     sp1.centre.z = Clavier.saisirDouble();
     Ecran.afficherln("Deuxieme sphere");
     Ecran.afficher("Rayon                    ? ");
     sp2.rayon = Clavier.saisirDouble();
     Ecran.afficher("Coordonnee x du centre ? ");
     sp2.centre.x = Clavier.saisirDouble();
     Ecran.afficher("Coordonnee y du centre ? ");
     sp2.centre.y = Clavier.saisirDouble();
     Ecran.afficher("Coordonnee z du centre ? ");
     sp2.centre.z = Clavier.saisirDouble();
     dx = sp1.centre.x-sp2.centre.x;
     dy = sp1.centre.y-sp2.centre.y;
     dz = sp1.centre.z-sp2.centre.z;
     distance = Math.sqrt(dx*dx+dy*dy+dz*dz);
     distance = distance-sp1.rayon-sp2.rayon;
      if ( distance < 0.0 ) {
       distance = 0.0; }
     Ecran.afficher("La distance est ",distance);
   }
}
```

Exemple d'exécution

IntersectionSphereDroite.java

```
/* Manipulations sur des spheres et des droites 3D */

public class IntersectionSphereDroite {

/* Type agrege de stockage d'une position 3D       */

   static class Position3D {
     double x = 0.0;
     double y = 0.0;
     double z = 0.0; };

/* Type agrege de stockage d'une sphere 3D         */

   static class Sphere {
     double rayon = 1.0;
     Position3D centre = new Position3D(); };

/* Type agrege de stockage d'une droite 3D         */

   static class Droite3D {
     double ax = 1.0;
     double bx = 0.0;
     double ay = 0.0;
     double by = 0.0;
     double az = 0.0;
     double bz = 0.0; };

/* Programme principal                             */

   public static void main(String [] args) {
     Sphere sp = new Sphere();
     Droite3D dr = new Droite3D();
     double dx;
     double dy;
     double dz;
     double a;
     double b;
     double c;
     double delta;
     boolean intersection;
    Ecran.afficherln("Sphere");
    Ecran.afficher("Rayon                    ? ");
    sp.rayon = Clavier.saisirDouble();
    Ecran.afficher("Coordonnee x du centre ? ");
    sp.centre.x = Clavier.saisirDouble();
    Ecran.afficher("Coordonnee y du centre ? ");
    sp.centre.y = Clavier.saisirDouble();
    Ecran.afficher("Coordonnee z du centre ? ");
    sp.centre.z = Clavier.saisirDouble();
    Ecran.afficherln("Deuxieme sphere");
```

```
    Ecran.afficherln("Droite");
    Ecran.afficher("Coefficient ax          ? ");
    dr.ax = Clavier.saisirDouble();
    Ecran.afficher("Coefficient bx          ? ");
    dr.bx = Clavier.saisirDouble();
    Ecran.afficher("Coefficient ay          ? ");
    dr.ay = Clavier.saisirDouble();
    Ecran.afficher("Coefficient by          ? ");
    dr.by = Clavier.saisirDouble();
    Ecran.afficher("Coefficient az          ? ");
    dr.az = Clavier.saisirDouble();
    Ecran.afficher("Coefficient bz          ? ");
    dr.bz = Clavier.saisirDouble();
    dx = sp.centre.x-dr.bx;
    dy = sp.centre.y-dr.by;
    dz = sp.centre.z-dr.bz;
    a = dr.ax*dr.ax + dr.ay*dr.ay + dr.az*dr.az;
    b = 2.0*(dr.ax*dx + dr.ay*dy + dr.az*dz);
    c = dx*dx + dy*dy + dz*dz - sp.rayon*sp.rayon;
    delta = b*b - 4.0*a*c;
     if ( delta >= 0.0 ) {
      intersection = true; }
       else {
      intersection = false; }
    Ecran.afficherln("Intersection : ",intersection);
  }
}
```

Exemple d'exécution

Exercice supplémentaire n°2

On souhaite développer un jeu de guerre en ambiance "Seconde guerre mondiale".
Pour ce faire, les blindés disponibles doivent être modélisés informatiquement.
Pour chacun d'eux, on souhaite gérer les caractéristiques suivantes:

- nom,
- type,
- pays,
- date,
- poids,
- hauteur,
- largeur,
- longueur caisse seule,
- longueur totale avec canon,
- équipage,
- caractéristiques d'armement,

- caractéristiques de blindage,
- caractéristiques de motorisation,
- caractéristiques de déplacement.

Les caractéristiques d'armement sont décrites de la manière suivante:

- caractéristiques de l'armement principal,
- caractéristiques de l'armement secondaire 1,
- caractéristiques de l'armement secondaire 2.

Les armements primaire et secondaires sont décrits par les points suivants:

- armement présent,
- nom,
- nombre de coups stockables,
- cadence de tir,
- létalité sur les fantassins,
- létalité sur les blindés.

Les caractéristiques de blindage sont:

- résistance du blindage de la caisse,
- résistance du blindage de la tourelle.

Les résistance de blindage sont (réels de 0.0 à l'infini):

- résistance blindage face avant,
- résistance blindage face latérale,
- résistance blindage face arrière,
- résistance blindage toit.

Les caractéristiques de motorisation sont:

- nom,
- type,
- cylindrée,
- puissance.

Les caractéristiques de déplacement sont:

- vitesse maximale sur route,
- capacité de franchissement en tout-terrain:
  - pente,
  - tranchée,
  - gué,
  - pression au sol,
- vitesse maximale en tout-terrain,
- autonomie sur route,
- autonomie en tout-terrain.

Exemple: Le char JS-2

- Nom: "JS-2 modèle 1944"
- Type: "Char lourd"

- Pays: "URSS"
- Date: 01/01/1944
- Poids: 46 tonnes
- Hauteur: 2.70 m
- Largeur: 3.10 m
- Longueur caisse seule: 6.80 m
- Longueur totale avec canon: 9.80 m
- Equipage: 4
- Armement principal, présent: oui
- Armement principal, nom: "122 mm D-25T L/43"
- Armement principal, nombre de coups stockables: 28 projectiles
- Armement principal, cadence de tir: 3 coups/mn
- Armement principal, létalité sur les fantassins: 10.0
- Armement principal, létalité sur les blindés: 20.0
- Armement secondaire 1, présent: oui
- Armement secondaire 1, nom: "Mitrailleuse DT 7.62 mm"
- Armement secondaire 1, nombre de coups stockables: 2330 projectiles
- Armement secondaire 1, cadence de tir: 550 coups/mn
- Armement secondaire 1, létalité sur les fantassins: 15.0
- Armement secondaire 1, létalité sur les blindés: 0.2
- Armement secondaire 2, présent: oui
- Armement secondaire 2, nom: "Mitrailleuse DShk 12.7 mm"
- Armement secondaire 2, nombre de coups stockables: 945 projectiles
- Armement secondaire 2, cadence de tir: 350 coups/mn
- Armement secondaire 2, létalité sur les fantassins: 18.0
- Armement secondaire 2, létalité sur les blindés: 0.4
- Résistance de blindage caisse, face avant: 120.0
- Résistance de blindage caisse, face latérale: 90.0
- Résistance de blindage caisse, face arrière: 60.0
- Résistance de blindage caisse, toit: 20.0
- Résistance de blindage tourelle, face avant: 90.0
- Résistance de blindage tourelle, face latérale: 90.0
- Résistance de blindage tourelle, face arrière: 90.0
- Résistance de blindage tourelle, toit: 30.0
- Motorisation nom: "V-2-IS"
- Motorisation type: "12 cylindres en V - Diesel"
- Motorisation cylindrée: 38.9 litres
- Motorisation puissance: 520 cv
- Caractéristiques de déplacement, vitesse maximale sur route: 37 km/h
- Caractéristiques de déplacement, capacité de franchissement en tout-terrain, pente: 36°

- Caractéristiques de déplacement, capacité de franchissement en tout-terrain, tranchée: 2.5 m
- Caractéristiques de déplacement, capacité de franchissement en tout-terrain, gué: 1.3 m
- Caractéristiques de déplacement, capacité de franchissement en tout-terrain, pression au sol: 0.81 kg/cm^2
- Caractéristiques de déplacement, vitesse maximale en tout-terrain: 19 km/h
- Caractéristiques de déplacement, autonomie sur route: 220 km
- Caractéristiques de déplacement, autonomie en tout-terrain: 178 km

Concevoir et valider un type agrégé Java (structure de données) permettant de modéliser les caractéristiques d'un blindé.

TypeAgregeBlinde.java

```
/* Type agrege de stockage d'un blinde                */

static class Blinde {
   String nom = "";
   String type = "";
   String pays = "";
   Date date = new Date();
   double poids = 0.0;
   double hauteur = 0.0;
   double largeur = 0.0;
   double longueurCaisse = 0.0;
   double longueurTotale = 0.0;
   int equipage;
   Armement armement = new Armement();
   Blindage blindage = new Blindage();
   Motorisation motorisation = new Motorisation();
   Deplacement deplacement = new Deplacement(); };

/* Type agrege de stockage d'une date                 */

static class Date {
   int jour = 1;
   int mois = 1;
   int annee = 1901; };

/* Type agrege de stockage d'un armement de blinde */

static class Armement {
   Arme armePrincipale = new Arme();
   Arme armeSecondaire1 = new Arme();
   Arme armeSecondaire2 = new Arme(); };
```

```
/* Type agrege de stockage d'une arme                */

static class Arme {
   boolean present = false;
   String nom = "";
   int nombreCoups = 0;
   int cadenceTir = 0;
   double letaliteSurFantassins = 0.0;
   double letaliteSurBlindes = 0.0; };

/* Type agrege de stockage d'un blindage de blinde */

static class Blindage {
   BlindageElement blindageCaisse = new BlindageElement();
   BlindageElement blindageTourelle = new BlindageElement(); };

/* Type agrege de stockage des caracteristiques    */
/* du blindage d'un element de structure de blinde */

static class BlindageElement {
   double avant = 0.0;
   double lateral = 0.0;
   double arriere = 0.0;
   double toit = 0.0; };

/* Type agrege de stockage d'une motorisation      */

static class Motorisation {
   String nom = "";
   String type = "";
   double cylindree = 0.0;
   double puissance = 0.0; };

/* Type agrege de stockage de caracteristiques     */
/* de deplacement                                  */

static class Deplacement {
   double vitesseMaxSurRoute = 0.0;
   Franchissement franchissement = new Franchissement();
   double vitesseMaxToutTerrain = 0.0;
   double autonomieSurRoute = 0.0;
   double autonomieToutTerrain = 0.0; };

/* Type agrege de stockage de caracteristiques     */
/* de franchissement                               */

static class Franchissement {
   double penteMax = 0.0;
   double trancheeMax = 0.0;
```

```
    double gueMax = 0.0;
    double pressionAuSol = 0.0; };
```

Exemple d'exécution

Auteur: Nicolas JANEY
UFR Sciences et Techniques
Université de Besançon
16 Route de Gray, 25030 Besançon
nicolas.janey@univ-fcomte.fr