**UB2/Faculté Math & Info/Département Informatique Année 2018/2019,**

**Filière/Module : 2ème LMD ACA/P.O.O Date 31/01/2019**

**Enseignant : N.Kadache Durée : 1 h 30 mn**

**Examen Final 2019**

**Questions de Cours (2.5X4 =10 Pts):**

Complétez les phrases suivantes :

**P1)** Une classe est une …………….. d’un ensemble d’…………….. ayant la même……………………………………………. et le même ………………. ?

**p2)** La …………………. des méthodes signifie l’existence d’un ensemble de méthodes dans une classe avec des paramètres différents en nombre et/ou en type. Une méthode ………………………. est une méthode reprise dans une classe héritière avec une implémentation différente

**p3)** Un attribut protégé en java (protected) est un attribut qu’on peut accéder dans ……………………………………………………..

**p4)** Une méthode …………………. Est une méthode polymorphe, un objet polymorphe est un objet qui ………………………………………………………………

**Exercice (4+4+2=10Pts):**

1. Donnez le code d’une classe Java nommée **Point3D** et caractérisée par :
   1. Trois attributs privés de type double x,y et z (à encapsuler)
   2. Un constructeur par défaut.
   3. Un constructeur paramétré.
   4. Une méthode **double distance(Point3D p)** qui renvoi la distance entre l’objet point en cours et le point p
2. Donnez le code d’une classe Java nommée **Sphere** caractérisée par :
   1. Un Point (objet de la classe précédente) qui représente le centre de la sphère (à encapsuler).
   2. Un attribut rayon de type double (à encapsuler)
   3. Un constructeur avec quatre paramètres représentant respectivement les coordonnées du point centre et le rayon du cercle.
   4. Une méthode **boolean intersect(Sphere s)** qui renvoi true si l’intersection de la sphère en cours avec s n’est pas vide.
3. Une boule est une sphère dans l’espace à n dimensions (n entier >0). Proposez un code pour la classe **Boule** caractérisée par :
   1. Un Point centre de cordonnées de type double (x0,x1,x2,…,xn).
   2. Un attribut rayon de type double.
   3. Un constructeur avec deux paramètres représentant respectivement les coordonnées du point centre (un tableau) et le rayon du cercle.
   4. Une méthode **boolean intersect(Boule b)** qui renvoi true si l’intersection de la boule en cours avec b n’est pas vide.

Indication : Ajoutez la dimension n comme attribut dans la classe Boule.

**N.B :**

1. La méthode Math.sqrt(x) renvoi la racine carrée de x.
2. La distance entre deux points dans l’espace 3D p1(x1,y1,z1) et p2 (x2,y2,z2) est donnée par
3. La distance entre deux points dans l’espace à dimension n p1(x0,x1…xn) et p2(y0,y1…yn) est donnée par .
4. Si **tab** est un tableau java alors **tab**.**length** représente le nombre d’éléments du tableau

Bon courage

**UB2/Faculté Math & Info/Département Informatique Année 2018/2019,**

**Filière/Module : 2ème LMD ACA/P.O.O Date 31/01/2019**

**Enseignant : N.Kadache Durée : 1 h 30 mn**

**Corrigé-type 2019**

**P1)** Une classe est une abstraction d’un ensemble d’objets ayant la même structure d’attributs et le même comportement ?

**p2)** La surcharge des méthodes signifie l’existence d’un ensemble de méthodes dans une classe avec des paramètres différents en nombre et/ou en type. Une méthode redéfinie est une méthode reprise dans une classe héritière avec une implémentation différente

**p3)** Un attribut protégé en java (protected) est un attribut qu’on peut accéder dans sa classe et toutes les classes héritières (dérivées) de sa classe.

**p4)** Une méthode redéfinie Est une méthode polymorphe, un objet polymorphe est un objet qui comporte au moins une méthode polymorphe.

**Exercice (4+4+2=10Pts):**

package examen;

/\*\*

\*

\* @author nabil

\*/

1- public class **Point3D** {

private double x;

private double y;

private double z;

public Point3D() {

}

public Point3D(double x, double y, double z) {

this.x = x;

this.y = y;

this.z = z;

}

public double distance(Point3D p){

return Math.sqrt((p.x-x)\*(p.x-x)+(p.y-y)\*(p.y-y)+(p.z-z)\*(p.z-z));

}

public double getZ() { return z; }

public void setZ(double z) { this.z = z; }

public double getY() { return y; }

public void setY(double y) { this.y = y; }

public double getX() { return x; }

public void setX(double x) { this.x = x; }

}

2- public class **Sphere** {

private Point3D centre;

private double rayon;

public Point3D getCentre() { return centre; }

public void setCentre(Point3D centre) { this.centre = centre; }

public double getRayon() { return rayon; }

public void setRayon(double rayon) { this.rayon = rayon; }

public Sphere(double x,double y,double z,double rayon) {

centre=new Point3D(x,y,z);

this.rayon=rayon;

}

public boolean intersect(Sphere b){

double bdist=centre.distance(b.getCentre());

double sommerayon=rayon+b.getRayon();

if(bdist<sommerayon) return true;

else return false;

}

}

3- public class Boule {

private double rayon;

private int dimension;

private double[] centre;

public Boule(double[] centre, double rayon) {

this.dimension=centre.length;

this.rayon = rayon;

this.centre = centre;

}

public boolean intersect(Boule b){

if(dimension!=b.dimension)

{

System.out.println("Erreur: les deux boules n'ont pas la même dimension");

return false;

}

else {

double dist=0;

for(int i=0;i<dimension;i++)

dist=dist+(b.getCentre()[i]-centre[i])\*(b.getCentre()[i]-centre[i]);

dist=Math.sqrt(dist);

if(dist<=rayon+b.getRayon()) return true;

else return false;

}

}

public double[] getCentre() { return centre; }

public void setCentre(double[] centre) { this.centre = centre; }

public double getRayon() { return rayon; }

public void setRayon(double rayon) { this.rayon = rayon; }

public int getDimension() { return dimension; }

}

/\*\*\*\*\*\*\*\*\*\*\*\*\* Test\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public class Examen {

public static void main(String[] args) {

Point3D p1=new Point3D(1,1,1);

Point3D p2=new Point3D(2,2,2);

System.out.println("distance entre p1 et p2:"+p1.distance(p2));

Sphere s1=new Sphere(p1.getX(),p1.getY(),p1.getZ(),1.0);

Sphere s2=new Sphere(p2.getX(),p2.getY(),p2.getZ(),1.0);

System.out.println("la spheres s1 interrsecte s2?:"+s1.intersect(s2));

s2.setRayon(0.5);

System.out.println("la spheres s1 interrsecte s2?:"+s1.intersect(s2));

double[] c1={1.0,1.0,1.0,1.0};

Boule b1 = new Boule(c1,1.0);

double[] c2={2.0,2.0,2.0,2.0};

Boule b2 = new Boule(c2,2.0);

System.out.println("intersection des deux boules?:"+b1.intersect(b2));

b2.setRayon(0.5);

System.out.println("intersection des deux boules?:"+b1.intersect(b2));

}

}