UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

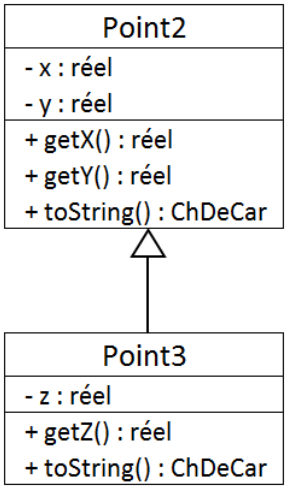
Licence INFO
Info0201

Travaux dirigés n° 7

Typage dynamique - polymorphisme

Exercice 1 (Cas d'école)

On considère deux classes **Point2** et **Point3**, représentant des points cartésiens en dimension 2 et 3 respectivement. Comme le montre le diagramme UML, on considère que les **Point3** sont des **Point2** dotés d'une troisième coordonnée (*cote*). Par ailleurs chacune de ces classes est dotée d'un unique constructeur, par initialisation.



1°) Donnez les codes des classes **Point2** et **Point3**.

2°) Soient les déclarations et instanciations suivantes :

```
Point2 ref2 = new Point2(3.5,-2.6);
Point3 ref3 = new Point3(3,-7.4,-5.63);
```

Peut-on réaliser les appels suivants ?

```
ref2.getX();      ref2.getZ();
ref3.getX();      ref3.getZ();
```

3°) On considère le code suivant :

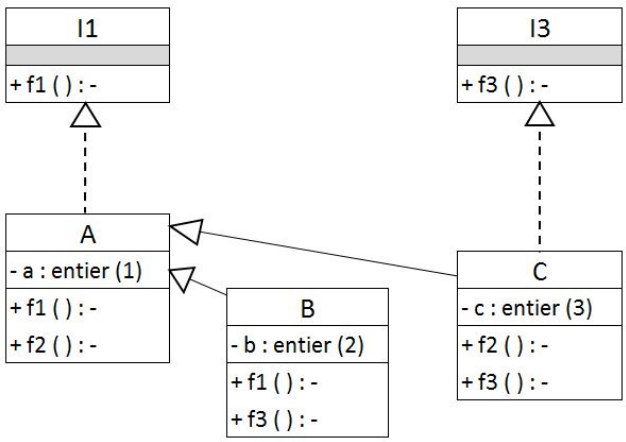
```
Point2[] tab = new Point2[10];
for (int i=0 ; i < tab.length ; i++)
    if ( Math.random() < 0.5 )
        tab[i] = new Point2(i,i);
    else tab[i] = new Point3(i,i,Math.random());
```

A la suite du code précédent on souhaite compter le nombre de **Point2** parmi les **Point3** du tableau `tab` :

- En ajoutant aux **Point2** un attribut donnant le type du Point créé (**Point2** ou **Point3** par exemple avec une énumération), on peut connaître le type de l'objet : modifiez l'ensemble des codes pour réaliser ce comptage (classes **Point2** et **Point3** ; classe de test).
- Proposez une solution pour le faire sans être obligé de modifier les codes des classes **Point2** et **Point3**.
- De même, calculez la moyenne des cotes des **Point3** référencés dans ce tableau.

Exercice 2 (Généricité)

On considère le diagramme de classes suivant :



1°) Est-il nécessaire / cohérent d'indiquer que la classe **B** implémente l'interface **I1** ? que **C** implémente **I1** ?

2°) Avec

```
I1 x = new B();
I3 y = new C();
```

- Expliquez pourquoi ce code est cohérent.
- Peut-on alors réaliser l'instruction `x = y` ? Pourquoi ?
  Si non, donnez un moyen de rendre cette instruction possible.

3°) Soit

```
I1 ref1A = new A();
I1 ref1B = new B();
I3 ref3C = new C();
```

Pour chacun des appels suivants, indiquez
- s'il est possible
- si oui, quelle est LA méthode appelée
- si non, expliquez pourquoi et [si possible] expliquez comment le corriger.

`ref1A.f1();` `ref1A.f2();` `ref1A.f3();` `ref1B.f1();` `ref1B.f2();` `ref1B.f3();`
`ref3C.f1();` `ref3C.f2();` `ref3C.f3();`

Exercice 3 (Keskimarsh)

On considère les trois classes suivantes ainsi que la classe de test :

```
public class C1{
   public int a;
   protected int b;
   private int c;
   protected static int d=0;

   private C1(){
      a=0;  b=2;  c=0;  d++;
   }

   public C1(int n){
      this();
      a = n;
   }

   public void setC(int a){
      this.c = 2*a;
   }

   public void add(int u){
      b += u;
   }

   public void affiche(){
      System.out.println("C1"
         + " (" + a + "," + b + ","
         + c + ") : " + d);
   }
}
```

```
public class C2 extends C1{
   public C2(int n){
      super(n);
      c=n+8;
   }
   public C2(){
   }

   public void affiche(){
     System.out.println("C2");
   }
}
```

```
public class C3 extends C1{
   private static double age = 12.5;

   public C3(){
      super(4);
   }

   public void affiche(){
      System.out.println("C3");
   }

   public static double getAge(){
      return age;
   }
}
```

```
public class Keskimarsh{
   public static void main(String[] d){
      C1 c11 = new C1(3);
      C1 c11b = new C1();
      C1 c13 = new C3();
      C3 c33 = new C3();
      C3 c31 = new C1(0);
      C2 c22 = new C2();
      C1 c12 = new C2(8);

      c13.affiche();
      c33.affiche();

      c13.getAge();
      c13.setC(int 4);
      c13.setC(4);
   }
}
```

- Les classes **C1** et **C3** ne contiennent pas d'erreur ; corrigez celles de **C2**.
- Pour chaque instruction de la classe **Keskimarsh**, précisez ses effets si elle est valide ; si elle ne l'est pas, justifiez pourquoi.