Université de Tlemcen
Faculté des Sciences
Département d'Informatique

Année universitaire 2019-2020
2ème année licence informatique
Programmation Orientée Objet

TP N° 3

Exercice 1

Soient les classes suivantes :

class A {
int a;
A(int a) { a=a; }
}
	class B extends A {
	int b;
	B(int b) { b=b; }
	}
		class C extends B {
		int c ;
		}

Corrigez les erreurs **sur les constructeurs** des classes A, B et C.

Exercice 2

Ecrire en java chacune des classes décrites ci-dessous :

1. Une **Personne** a un nom (String) et un nombre d'enfants (int).
2. Un **Salarié** est une personne qui perçoit un salaire (double). Tout salarié, a droit à une allocation familiale "AF" (int) de 300 DA pour chacun de ses enfants. "primeAF ()" permet de calculer le total des AFs que perçoit le salarié. "afficher()" permet d'afficher son nom et sa prime.
3. Le programme **TestAF** permet de créer un salarié et d'afficher son nom et sa prime.

Exercice 3

Soit la classe **Point** ainsi définie :

```
class Point {
    private double x ;
    private double y ;
    public Point (double abs, double ord) { x = abs ; y = ord ; }
    public void affiche (){ System.out.print (" (" + x + "," + y+ ")") ; }
    public boolean identique (Point a) { return ( (a.x==x) && (a.y==y) ) ; }
}
```

1- Réaliser une classe **PointNom**, dérivée de Point
   a. Définir un attribut nom (char) permettant de manipuler des points avec leurs noms
   b. Ecrire un constructeur PointNom(double x, double y, char nom)
   c. Redéfinir (compléter) la méthode affiche() afin qu'elle affiche en plus des coordonnées le nom du point, par exemple : M(5,2)
   d. Dans la classe PointNom, redéfinir la méthode identique fournissant la valeur true lorsque les deux points concernés ont à la fois mêmes coordonnées et même nom

2- Soit la classe **TestPoint**. Complétez et répondez aux commentaires du code suivant :

```
public class TestPoint{
        public static void main (String args[]){
                Point p1, p2;
                p2= new PointNom(2,5,'A') ;
                // Affichez seulement les coordonnes de p2  "(2, 5)", sans le nom.

                p1=p2;
                System.out.println(p2.identique(p1));          // expliquez l'affichage

                //changez le nom de p1 par 'B'

                System.out.println(p2.identique(p1));          // expliquez l'affichage

                p1 = new PointNom(2,5,'A');
                System.out.println(p2.identique(p1)+ "  "+p1+"  "+p2);
                // Expliquez l'affichage.
                //Que faut-il faire pour que « p2. identique(p1) » retourne true ?
        }
}
```

Exercice 1

Explication

Pour A : ajouter this dans le constructeur de A afin de lever **l'ambigüité** entre **attribut** et **paramètre a**.
Pour B : java appelle le constructeur super() sans paramètre qui n'existe pas, puisque dans A, un constructeur avec un paramètre a été défini.
Pour C : il faut définir explicitement un constructeur, puisque java va tenter de vous fournir un constructeur sans paramètre, cependant il fera appel à super() sans paramètre inexistant.

Une solution

```
class A {
   int a;
   A(int a){ this.a=a;}
}
-----------------
class B extends A {
  int b;
  B(int a, int b){ super(a); this.b=b;}
}
-------------------
class C extends B {
   int c ;
   C(int a, int b, int c){    super(a,b);this.c=c;  }
}
```

Exercice 2

```
class Personne {
   String nom ;
   int nombreEnfants;

   Personne(String n, int ne){ nom = n ; nombreEnfants = ne ;}

   String getNom(){return nom;}
}
----------------------
class Salarie extends Personne {
    int salaire ;
    static int AF = 300;
    Salarie (String n, int ne, int s) { super(n,ne) ; salaire = s ; }
    int primeAF (){ int taux = nombreEnfants * AF ;        return taux ;}
    void afficher(){System.out.println(getNom()+" "+primeAF());}
}
----------------------
class TestAF{
     static public void main(String[] sArgs){
         Salarie s=new Salarie("Omar", 4, 2000);
         s.afficher();
     }
}
```

Exercice 3

```
class PointNom extends Point{
    char nom ;
   public PointNom (double x, double y, char nom){
      super (x, y) ;
      this.nom = nom ;
   }

   public void affiche(){
      System.out.print (nom);super.affiche();
   }

   public boolean identique (Point a){ System.out.print (nom);
      return ( (super.identique(a)) &&   (((PointNom)a).nom==nom) ) ;
   }
}
-------------------------------------------------
public class TestPoint{
   public static void main (String args[]){
      Point p1, p2;
      p2= new PointNom(2,5,'A') ;
      ((Point) p2).affiche() ; // Affichez seulement les coordonnes  "(2, 5)"
      p1=p2;
      System.out.println(p2.identique(p1));   // affichage ? expliquez.
         // true car c'est la méthode la plus spécifique qui sera appelée.
         // Aussi, p et pn1 pointent sur la même adresse (rééférence).
      ((PointNom)p1).nom ='B';  //changez le nom de p1
      System.out.println(p2.identique(p1)); //Affichage ? Expliquez.
         // Puisque p1 et p2 référencent le même objet, le nom de p2 est aussi 'B'
         // System.out.println(((PointNom)p2).nom); //pour vérifier.
      p1 = new PointNom(2,5,'A'); //nouvelle référence pour p1
      System.out.println(p2.identique(p1)+ "  "+p1+"  "+p2);
         // affiche false parceque le nom de p2 est maintenant 'B'
         //p1 et p2 référencent deux objets différents
   }
}
```