Université de Tlemcen
Faculté des Sciences
Département d'Informatique

Année universitaire 2018-2019
2ème année licence informatique
Programmation Orientée Objet

Complément du TD N° 1

Exercice n° 6

```
class A {
        public static void main (String [] args){
        A a,b,c ;
        a=new A() ;     b=new A() ;             c=b ;       a=b ;
        }
}
```

Combien d'instances de la classe A sont créées pendant l'exécution du code suivant ?
Combien en reste après le passage du "Garbage collector" ?

Deux (02) instances.
Une seule, l'instance "new A()" n'étant plus référencée, sera supprimée par le GC

Exercice n° 7

```
class D {
                public static int x ;
                public int y ;
                public static travailler() {x++;}
                public D() {x++; y-- ; }
}
```

Qu'affichera le code suivant ?

```
D.travailler() ; D a=new D() ; D b=new D() ; a.travailler() ;
System.out.println(b.x + " et " + b.y) ;
```

4 et -1
D.travailler() ; //incrémente x de 0 à 1
D a=new D() ; //incrémente x de 1 à 2 et décrémente y de a de 0 à -1
D b=new D() ; //incrémente x de 2 à 3 et décrémente y de b de 0 à -1
a.travailler() ; // incrémente x de 3 à 4

Exercice n° 8

```
class D {
                public int x ;
                public D() { x=3 ; } ;
                public D( int a){this() ; x=x+a ;} ;
                public D( int a, int b){this(b) ; x= x-a ;}
}
```

Qu'affichera le code suivant ?

```
D a=new D(5,6) ;
System.out.println(a.x) ;
```

D a=new D(5,6) ; // appelle this(b) qui lui-même appelle this() d'où x=3, puis x=x+6 d'où x=9, puis x=x-5 et donc a.x affiche 4

Exercice n° 9

```
class C {
        public static int i;
        public int j;
        public C() {i++; j=i; }
}
```

Qu'affichera le code suivant ?

```
C x=new C(); C y=new C(); C z= x;
System.out.println(z.i + " et " + z.j);
```

C x=new C(); // incrémente la variable de classe i de 0 à 1 et x.j=1

C x=new C(); // la variable de classe i de 1 à 2 et y.j=1

C z= x; // z reference la même instance que x

Affiche 2 et 1

Exercice n° 10

Développez en java chacune des classes décrites ci-dessous :

1- Une personne a un nom et un nombre d'enfants et une méthode getNom() qui retourne son nom.

2- Un salarié est une personne qui perçoit un salaire. Tout salarié, a droit à une Allocation Familiale (AF) 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.

```
class Personne {
    private String nom ;
    protected 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 n° 11

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- Ecrire une classe **TestPoint** qui permet de :
   a. créer un objet pn de PointNom nommé 'A' avec les coordonnées (2,5)
   b. afficher ses coordonnées
   c. créer un objet p de Point en lui affectant pn
   d. comparez p et pn avec la méthode identique() de PointNom et discutez le résultat.

```
class Point {
    private double x ; // abscisse
    private double y ; // ordonnee

    public Point (double abs, double ord) {
        x = abs ; y = ord ;
    }
    public void affiche (){
       System.out.println (" (" +x+","+y+")") ;
    }

    public boolean identique (Point a){
        return ( (a.x==x) && (a.y==y) ) ;
    }
}
```

```
class PointNom extends Point{
    private 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){
        return ( (super.identique((PointNom)a)) &&
                 (((PointNom)a).nom==nom) ) ;
    }
}
```

```
public class TsPointN{
   public static void main (String args[]){
      Point p ;
      PointNom pn1;
      Pn1= new PointNom(2,5,'A') ;
      pn1.affiche() ; // methode de PointNom
      p=pn1;
      System.out.println(pn1.identique(p)); // true car c'est la méthode la plus spécifique
            //qui sera appelée. p et pn1 référencent la même instance.
   }

     }
```