Université Tunis El-Manar	Faculté des Sciences de Tunis
Module : Programmation.O.O	**Année universitaire :** 2024-2025
Section : MI 2, PI 2	Hajer Dammak

Série N° 5 : Héritage

Points essentiels :

- Héritage et polymorphisme
- L'utilisation de `super`

Exercice 1 :

Déterminer ce qui est affiché par le programme suivant :

```
class Personne{
    void saluer (){
    System.out.println("Bonjour, comment ça va?");
    }
}
class Enseignant extends Personne{
}
class EnseignantSecondaire extends Enseignant{
    void saluer (){
    System.out.println("Bonjour, comment vont mes élèves?");
    }
class EnseignantUniversitaire extends Enseignant{
    void saluer (){
    System.out.println("Bonjour, comment vont mes étudiants?");
    }
}
class TestPersonne{
    public static void main(String[] args)
    {
    Personne p1 = new Personne();
    Personne p2 = new Enseignant ();
    Personne p3 = new EnseignantSecondaire ();
    EnseignantUniversitaire p4 = new EnseignantUniversitaire ();
    Enseignant p5 = new EnseignantSecondaire();
    Enseignant p6 = new Enseignant();
    p1.saluer();
    p2.saluer();
    p3.saluer();
    p4.saluer();
    p5.saluer();
    p6.saluer();
    }
}
```

Indication :

- La méthode `saluer()` est une méthode polymorphe (déclarée dans une super classe et redéfinie par une sous classe).
- Un objet peut être vu comme une instance de sa classe et aussi comme une instance de toute classe qui dérive de sa propre classe.

Exercice 2 :

Écrire les classes nécessaires au fonctionnement du programme suivant, en ne fournissant que les méthodes nécessaires à ce fonctionnement :

```
class TestMetiers{
    public static void main(String[] args)
    {
    Personne[] personnes = new Personne [4];
    personnes [0] = new Personne ("Salah");
    personnes [1] = new Forgeron ("Ali");
    personnes [2] = new Menuisier ("Mohamed");
    personnes [3] = new Forgeron ("Amor");

    for (int i=0; i<personnes.length; i++)
        personnes [i].affiche();
    }
}
```

On obtient sur écran :

```
Je suis Salah
Je suis Ali le forgeron
Je suis Mohamed le menuisier
Je suis Amor le forgeron
```

Indication :

- Une classe dérivée n'hérite pas du constructeur de sa classe mère (un constructeur n'est pas considéré comme un membre).

Exercice 3 :

Compléter les classes suivantes pour que la classe `Test` fonctionne correctement.

```
class Point (...)
class Cercle extends Point (...)
class Cylindre extends Cercle {...)

class Test (
public static void main(String[] args){
    /* Un point est défini par ses coordonnées x et y */
    Point p = new Point (2.0, 2.0);
    System.out.println("Point -> "+p);
    /* Un cercle est défini par les coordonnées
    x et y de son centre et par son rayon */
    Cercle c = new Cercle (2.0, 2.0, 3.0);
    System.out.println ("Cercle -> "+c+", surface :"+c.surface());
    /* Un cylindre est défini par les coordonnées
    x et y de son centre, rayon et par sa hauteur */
    Cylindre cc = new Cylindre (2.0, 2.0, 3.0, 10.0);
    System.out.println("Cylindre ->"+cc+ ", surface :" + cc.surface());
    }
}
```

On obtient sur écran :

```
Point-> (2.0, 2.0)
Cercle -> rayon : 3.0, centre : (2.0, 2.0), surface : 28.259
Cylindre -> hauteur: 10.0, rayon : 3.0, centre : (2.0, 2.0), surface: 244.92
```

Indication :

- Utiliser `super` chaque fois que cela est possible.
- Surface d'un cercle = 3.14×rayon×rayon.
- Surface d'un cylindre = 2×3.14×rayon×rayon + 2×3.14×rayon×hauteur (ou encore 2×surface d'un cercle +2×3.14×rayon×hauteur).
- On doit attribuer le modificateur `public` pour redéfinir la méthode `toString()` car on ne peut pas redéfinir une méthode en diminuant son niveau de visibilité.