**DUT Informatique – S2**

**Programmation Orientée Objet**
**TD & TP N°6**
Héritage (2ème partie)

---

# TD N°6 :

## Exercice 1

Soit la classe suivante :

```
public class Personne
{
     private String nom;
     private int    age;
     …

     public Personne(String unNom, int unAge)
     {
          this.nom  =  new String(unNom);
          this.age  =  unAge;
     }

     public boolean equals(Personne p)
     {
          return (this.nom.equals(p.nom) && this.age == p.age);
     }

} //fin classe Personne
```

Soit les instructions :

```
{
     (1) Personne p = new Personne("titi", 30);
     (2) Object   o = new Personne("titi", 30);

     (3) if(p.equals(o))
            (4)System.out.println("caractéristiques égales");
         else
            (5)System.out.println("caractéristiques différentes");
}
```

a) Dessiner les références, les instances et leurs liens.
b) Y-a-t-il des erreurs à la compilation ? Dans tous les cas justifier votre réponse.
c) Quelle instruction parmi (4) et (5) est exécutée ? Pourquoi ?
d) Que faut-il faire pour avoir le comportement attendu ?
e) Considérer la classe *Salarie* héritant de la classe *Personne* donnée en annexe.
   Écrire la méthode *equals* de *Salarie*.

## Exercice 2

On souhaite concevoir une classe *Club* sachant qu'un club est caractérisé par un nom et une liste d'adhérents qui sont des personnes étudiantes, salariées ou ni l'un ni l'autre.
En utilisant les classes *Personne*, *Etudiant* et *Salarie* fournies en annexe écrire une classe *Club* qui rend les services suivants (certaines parties sont volontairement masquées) :

public class **Club** extends java.lang.Object

### Constructor Summary

| |
|---|
| **Club**()<br>constructeur vide : ne fait rien |
| **Club**(Club c)<br>constructeur par copie |
| **Club**(String unNomClub, …[] desAdherents)<br>initialise le *Club* courant |

### Method Summary

| | |
|---|---|
| …[] | **getAdherents**()<br>retourne les tableau des adhérents du *Club* courant |
| String | **getNomClub**()<br>retourne le nom du *Club* courant |
| void | **init**()<br>initialise interactivement le *Club* courant |
| String | **toString**()<br>retourne la chaîne de caractères représentant le *Club* courant |

a) Écrire les 2 variables d'instances.
b) Écrire le constructeur vide et les méthodes d'accès
c) Écrire le constructeur *Club(String unNomClub, …[]desAdherents)* sachant que le tableau *desAdherents* contient les adhérents du *Club* que l'on est en train d'initialiser (ne pas faire de copie de ce tableau).
d) Écrire le constructeur par copie. Remarque : il faut récupérer le tableau des adhérents du *Club c*, en faire une copie, ainsi qu'une copie de chacune des instances qu'il réfère. Afin de comprendre le mécanisme de copie :
    i. dessiner le tableau d'adhérents de *c* (avec les références et les instances) et le tableau d'adhérents de *this* que l'on veut obtenir
    ii. en déduire les instructions nécessaires (vous pouvez utiliser l'opérateur **instanceof**)
e) Écrire la méthode *toString* qui retourne une chaîne contenant le nom du *Club* courant et les information disponibles sur chaque adhérent
f) Écrire la méthode *init( )* qui fonctionnera selon le mode suivant :
    i. demander le nombre d'adhérents à l'utilisateur
    ii. pour chacun d'eux : demander s'il est salarié, étudiant, ou ni l'un ni l'autre, créer et initialiser une instance en conséquence

# TP N°6 :

1- Finir le TP N°5
2- Écrire les classes *Personne.java*, *Salarie.java* et *Etudiant.java* de l'exercice 2 du TD N°6.
3- Écrire et tester la classe *Club* de l'exercice 2 du TD N°6.
4- Dans une classe *TestClub* créer et afficher un *Club c1* initialisée interactivement et un *Club c2* copie de celui référencé par *c1*.

# Annexes

public class **Personne**

## Constructor Summary

| |
|---|
| **Personne**()<br>constructeur vide (ne fait rien) |
| **Personne**(Personne p)<br>constructeur par copie |
| **Personne**(java.lang.String n, int a)<br>initialise la *Personne* courante avec nom, age |

## Method Summary

| | |
|---|---|
| boolean | **equals**(Object o) retourne *true* si la *Personne* o a les mêmes caractéristiques que la *Personne* courante |
| int | **getAge**() retourne l'age de la *Personne* courante |
| String | **GetNom**() retourne le nom de la *Personne* courante |
| void | **init**() initialise interactivement la *Personne* courante |
| String | **toString**() retourne la chaîne de caractères représentant la *Personne* courante |

public class **Salarie** extends Personne

## Constructor Summary

| |
|---|
| **Salarie**() constructeur vide |
| **Salarie**(Salarie s) constructeur par copie |
| **Salarie**(String unNom, int unAge, String unNumeroSecu, String unEmployeur)<br>initialise le *Salarie* courant |

## Method Summary

| | |
|---|---|
| boolean | **equals**(Object o) retourne *true* si le *Salarie* référencé par *o* les mêmes caractéristiques que le *Salarie* courant |
| String | **getEmployeur**() retourne l'employeur du *Salarie* courant |
| String | **getNumeroSecu**() retourne le numéro de sécurité sociale du *Salarie* courant |
| void | **init**() initialise interactivement le *Salarie* courant |
| String | **toString**() retourne la chaîne de caractères représentant le *Salarie* courant |

### Methods inherited from class Personne

getAge, getNom

public class **Etudiant** extends Personne

# Constructor Summary

| |
|---|
| **Etudiant**()<br>constructeur vide |
| **Etudiant**(Etudiant e)<br>constructeur par copie |
| **Etudiant**(jString unNom, int unAge, String unNumeroEtudiant, String uneFac)<br>initialise l'*Etudiant* courant |

# Method Summary

| | |
|---|---|
| boolean | **equals**(Object o)<br>retourne *true* si l'*Etudiant* référencé par *o* a les mêmes caractéristiques que l'*Etudiant* courant |
| String | **getFaculte**()<br>retourne la faculté où étudie l'*Etudiant* courant |
| String | **getNumeroEtudiant**()<br>retourne le numéro d'étudiant de l'*Etudiant* courant |
| void | **init**()<br>initialise interactivement l'*Etudiant* courante |
| String | **toString**()<br>retourne la chaîne de caractères représentant l'*Etudiant* courant |

| **Methods inherited from class Personne** |
|---|
| getAge, getNom |