

## TP7

### Objectifs :

- Maîtriser les concepts de base de la POO
- 

### Exercice 1 :

Dans un fichier « **salarie.class.php** », créer la classe **Salarié** qui possède les attributs suivants :

- **matricule** int
- **nomComplet** String
- **salaire** float
- **tauxCS** float (le même taux des cotisations sociales pour tous les salariés).

- 1- Définir la classe **Salarié** en interdisant l'accès aux attributs.
- 2- Ajouter un constructeur dans lequel on peut utiliser :
  1. Un constructeur par défaut et en même temps un constructeur paramétré qui peut initialiser tous ou une partie des attributs.
- 3- Ajouter les accesseurs et les mutateurs de chaque attribut.
- 4- Ajouter la méthode **afficher()** qui permet d'afficher les informations d'un salarié.
- 5- Ajouter la méthode **calculerSalaireNet()** qui retourne le salaire net d'un salarié :
  - a. **SalaireNet = salaire - (salaire \* tauxCS)**
- 6- Dans fichier, **main.php**, instancier un ensemble de salariés et afficher leurs informations et salaires nets. Indication : **Utiliser require\_once("salarie.class.php");** au début du fichier main.php

### Exercice 2 :

1. Définir une classe **Client** avec les attributs suivants : **CIN, Nom, Prénom, Tél.**
2. Définir à l'aide des propriétés les méthodes d'accès aux différents attributs de la classe.
3. Définir un constructeur permettant d'initialiser tous les attributs.
4. Définir la méthode **Afficher ( )** permettant d'afficher les informations du Client en cours.
5. Créer Une classe **Compte** caractérisée par son solde et un code qui est incrémenté lors de sa création ainsi que son **propriétaire** qui représente un client.

6. Définir à l'aide des propriétés les méthodes d'accès aux différents attributs de la classe (le numéro de compte et le solde sont en lecture seule)
7. Définir un constructeur permettant de créer un compte en indiquant son propriétaire.
8. Ajouter à la classe Compte les méthodes suivantes :
  - i. Une méthode permettant de **Crediter()** le compte, prenant une somme en paramètre.
  - ii. Une méthode permettant de **Crediter()** le compte, prenant une somme et un compte en paramètres, créditant le compte et débitant le compte passé en paramètres.
  - iii. Une méthode permettant de **Debiter()** le compte, prenant une somme en paramètre
  - iv. Une méthode permettant de **Débitier()** le compte, prenant une somme et un compte bancaire en paramètres, débitant le compte et créditant le compte passé en paramètres
  - v. Une méthode qui permet d'afficher le résumé d'un compte.
  - vi. Une méthode qui permet d'afficher le nombre des comptes créés.
9. Créer un programme de test pour la classe Compte.

### Exercice 3 :

- 1) Ecrire une classe Bâtiment avec les attributs suivants :
  - a. ✓ **adresse**
- 2) La classe Bâtiment peut être instanciée des deux façons suivantes :
  - b. ✓ **Batiment(),**
  - c. ✓ **Batiment (adresse).**
- 3) La classe Bâtiment doit contenir des **accesseurs et mutateurs (ou propriétés)** pour les différents attributs.
- 4) La classe Bâtiment doit contenir une méthode **ToString ()** donnant une représentation du Bâtiment.
- 5) Ecrire une classe Maison héritant de Bâtiment avec les attributs suivants :

- d. ✓ **NbPieces**: Le nombre de pièces de la maison.
- 6) La classe Maison peut être instanciée des deux façons suivantes :
- e. ✓ **Maison()**,
- f. ✓ **Maison(adresse, nbPieces)**.
- 7) La classe Maison doit contenir **des accesseurs et mutateurs (ou des propriétés)** pour les différents attributs.
- 8) La classe Maison doit contenir une méthode **ToString ()** donnant une représentation de la Maison.
- 9) Ecrire aussi un programme afin de tester ces deux classes.

#### Exercice 4 :

Soit les classes suivantes :

Une classe **Personne** qui comporte trois champs privés, **nom**, **prénom** et **date de naissance**.

Cette classe comporte un **constructeur** pour permettre **d'initialiser les données**. Elle comporte également une méthode polymorphe **Afficher** pour afficher les données de chaque personne.

Une classe **Employé** qui dérive de la classe **Personne**, avec en plus un champ **Salaire** accompagné de sa propriété, un **constructeur** et la **redéfinition de la méthode Afficher**.

Une classe **Chef** qui dérive de la classe **Employé**, avec en plus un champ **Service** accompagné de sa propriété, un **constructeur** et la **redéfinition de la méthode Afficher**.

Une classe **Directeur** qui dérive de la classe **Chef**, avec en plus un champ **Société** accompagné de sa propriété, un **constructeur** et la **redéfinition de la méthode Afficher**.



### Travail à faire :

- 1) Ecrire les classe **Personne**, **Employé**, **Chef** et **Directeur**.
- 2) Créer un programme de test qui comporte un tableau de huit personnes : cinq employés, deux chefs et un directeur (8 références de la classe **Personne** dans lesquelles ranger 5 instances de la classe **Employé**, 2 de la classe **Chef** et 1 de la classe **Directeur**).
- 3) Affichez l'ensemble des éléments du tableau.