

## TP 2 : Programmation objet avec PHP

---

### Exercice 1

Considérons une classe appelée `Point` ayant les attributs suivants :

- `$_abs` : un attribut privé de type float
- `$_ord` : un attribut privé de type float

1. Créez la classe `Point`
2. Définissez un constructeur avec deux paramètres `Point(float $_abs, float $_ord)`
3. Créez ou générez les getters et setters pour les deux attributs.
4. Définissez la méthode `__toString()` qui retourne la représentation mathématique d'un point : `($_abs, $_ord)`.
5. Écrivez la méthode `calculerDistance(Point $p)` qui permet de calculer la distance entre le point de l'objet courant (`$this`) et l'objet `$p` passé en paramètre. Nous rappelons que la distance entre deux points  $A(x_1, y_1)$  et  $B(x_2, y_2)$ , en mathématiques, est égale à :  
$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$
6. Écrivez la `calculerMilieu(Point $p)` qui permet de calculer et de retourner un objet correspondant au milieu du segment défini par le point de l'objet courant (`$this`) et l'objet `$p` passé en paramètre. Nous rappelons que les coordonnées d'un point  $M(x_M, y_M)$  milieu de  $A(x_1, y_1)$  et  $B(x_2, y_2)$ , en mathématiques, sont :

- $x_M = \frac{x_1 + x_2}{2}$
- $y_M = \frac{y_1 + y_2}{2}$

La méthode doit retourner un objet `Point` et pas les coordonnées.

Considérons maintenant une deuxième classe appelée `TroisPoints` ayant les attributs suivants :

- `$_premier` : un attribut privé de type `Point`
  - `$_deuxième` : un attribut privé de type `Point`
  - `$_troisième` : un attribut privé de type `Point`
7. Créez ou générez les getters/setters et le constructeur avec trois paramètres de la classe `TroisPoints`.
  8. Écrivez une méthode `sontAlignes()` qui retourne `true` si les trois points `$_premier`, `$_deuxième` et `$_troisième` sont alignés, `false` sinon. Nous rappelons que trois points A, B et C sont alignés si  $AB = AC + BC$ ,  $AC = AB + BC$  ou  $BC = AC + AB$  (AB désigne la distance séparant le point A du point B, pareillement pour AC et BC).
  9. Écrivez une méthode `estIsocèle()` qui retourne `true` si les trois points `$_premier`, `$_deuxième` et `$_troisième` forment un triangle isocèle, `false` sinon. Nous rappelons qu'un triangle ABC est isocèle si  $AB = AC$  ou  $AB = BC$  ou  $BC = AC$ .

10. Dans un fichier `index.php`, demandez à l'utilisateur de saisir des coordonnées pour trois points. Ensuite, utilisez les classes et les méthodes précédentes pour afficher tous les détails sur ces trois points, les milieux, les distances qui les séparent, afficher s'ils sont alignés, s'ils forment un triangle isocèle...

## Exercice 2

Considérons une classe appelée **Stagiaire** ayant les attributs suivants :

- `$_nom` : un attribut privé de type chaîne de caractère
- `$_notes` : un attribut privé de type tableau de réels

1. Créez la classe **Stagiaire**
2. Créez ou générez les getters et setters des deux attributs.
3. Définissez un constructeur avec deux paramètres `Stagiaire(string $_nom, array $_notes)`
4. Écrivez la méthode `calculerMoyenne()` qui permet de retourner la moyenne des notes d'un stagiaire
5. Écrivez les méthodes `trouverMax()` et `trouverMin()` qui permettent de retourner respectivement les notes max et min d'un stagiaire.

Considérons maintenant une classe appelée **Formation** ayant les attributs suivants :

- `$_intitulé` : un attribut privé de type chaîne de caractère
- `$_nbrJours` : un attribut privé de type entier
- `$_stagiaires` : un tableau d'objets de type **Stagiaire**

6. Créez la classe **Formation**, générez les getters et setters de ses attributs, et définissez le constructeur `Formation(string $_intitulé, int $_nbrJours, array $_stagiaires)`
7. Écrivez une méthode `calculerMoyenneFormation()` qui retourne la moyenne d'un objet de type formation (la moyenne des moyennes des stagiaires)
8. Écrivez une méthode `getIndexMax()` qui retourne l'indice du stagiaire dans le tableau `stagiaires` ayant la meilleure moyenne de la formation.
9. Écrivez une méthode `afficherNomMax()` qui affiche le nom du premier stagiaire ayant la meilleure moyenne d'une formation.
10. Écrivez une méthode `afficherMinMax()` qui affiche la note minimale du premier stagiaire ayant la meilleure moyenne d'une formation.
11. Écrivez une méthode `trouverMoyenneParNom(string $nom)` qui affiche la moyenne du premier stagiaire dont le nom est passé en paramètre.
12. Dans un fichier `index.php`, testez toutes les méthodes réalisées dans les questions précédentes (créez par exemple trois objets **Stagiaire** et affectez les à une même formation et faites appel aux quatre dernières méthodes que vous avez implémentées).

## Exercice 3

Considérons les deux classes **Personne** et **Adresse**. Les attributs de la classe **Adresse** sont :

- `$_rue` : un attribut privé de type chaîne de caractères.
- `$_ville` : un attribut privé de type chaîne de caractère.
- `$_codePostal` : un attribut privé de type chaîne de caractère.

Les attributs de la classe **Personne** sont :

- `$_nom` : un attribut privé de type chaîne de caractère.
- `$_sexe` : un attribut privé de type caractère (cet attribut aura comme valeur soit 'M' soit 'F').
- `$_adresses` : un attribut privé de type tableau d'objet de la classe **Adresse**.

1. Créez les deux classes **Adresse** et **Personne** dans deux fichiers séparés. N'oubliez pas de créer/générer les getters/setters et le(s) constructeur(s).
2. Créez une troisième classe **ListePersonnes** ayant un seul attribut `$_personnes` : un tableau de maximum 10 objets de la classe **Personne**. Générez les getters/setters et le constructeur de cette classe.
3. Écrivez la méthode `findByNom(string $s)` qui permet de chercher dans le tableau `$_personnes` si un objet dont le nom égal au paramètre `s` existe. Si c'est le cas, elle retourne le premier objet correspondant, sinon `null`.
4. Écrivez la méthode `findByCodePostal(string $cp)` qui permet de vérifier dans le tableau `$_personnes` si un objet possède au moins une adresse dont le code postal égal au paramètre `$cp`. Si c'est le cas, elle retourne `true`, sinon `false`.
5. Écrivez la méthode `countPersonneVille(string $ville)` qui permet de calculer le nombre d'objets dans le tableau `$_personnes` ayant une adresse dans la ville passée en paramètre.
6. Écrivez la méthode `editPersonneNom(string $oldNom, string $newNom)` qui remplace les noms de personnes ayant un nom égal à la valeur `$oldNom` par `$newNom`.
7. Écrivez la méthode `editPersonneVille(string $nom, string $newVille)` qui remplace les villes de personnes ayant un nom égal à la valeur du paramètre `$nom` par `$newVille`.
8. Dans un fichier `index.php`, testez toutes les méthodes réalisées dans les questions précédentes.