

TP 1 : Classes/objets avec Java

Exercice 1

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

- `abs` : un attribut privé de type `double`
- `ord` : un attribut privé de type `double`

1. Créez la classe `Point`
2. Générez les getters et setters pour les deux attributs.
3. Définissez la méthode `toString()` qui retourne la représentation mathématique d'un point : (`abs`, `ord`).
4. Définissez un constructeur avec deux paramètres `Point(double abs, double ord)`
5. Définissez une méthode `calculerDistance(Point p)` qui calcule et retourne 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 :

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Utiliser la méthode statique `sqrt(double a)` de la classe `Math`

6. Définissez une méthode statique `distance(Point p1, Point p2)` qui calcule et retourne la distance entre les deux points passés en paramètres.
7. Définissez une méthode `calculerMilieu(Point p)` qui calcule et retourne un objet de type `Point` 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`
8. Générez les getters/setters et le constructeur avec trois paramètres de la classe `TroisPoints`.
 9. 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).

10. Écrivez une méthode `estIsocele()` 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 $BC = AC$.
11. Dans la méthode `main()` de la classe principale `Main`, demandez à l'utilisateur de saisir les coordonnées de 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 Java appelée `Stagiaire` ayant les attributs suivants :

- `nom` : un attribut privé de type chaîne de caractère
 - `notes` : un attribut privé de type tableau de réels (`float[] notes`)
1. Créez la classe `Stagiaire`
 2. Générez les getters et setters des deux attributs.
 3. Définissez un constructeur avec deux paramètres `Stagiaire(String nom, float[] 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, Stagiaire [] 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 `trouverIndiceMax()` qui retourne l'indice du stagiaire dans le tableau `stagiaires` ayant la meilleure moyenne de la formation.
 9. Écrivez une méthode `trouverNomMax()` qui retourne le nom du premier stagiaire ayant la meilleure moyenne d'une formation.
 10. Écrivez une méthode `trouverMinMax()` qui retourne la note minimale du premier stagiaire ayant la meilleure moyenne d'une formation.
 11. Écrivez une méthode `trouverMoyenneParNom(String nom)` qui retourne la moyenne du premier stagiaire dont le nom est passé en paramètre.
 12. Dans la méthode `main` de la classe principale `Main`, 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 caractère.
- **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 générer les getters/setters et le(s) constructeur(s).
 2. Créez une troisième classe **ListePersonnes** ayant un seul attribut **personnes** : un tableau d'objets de ltype **Personne**. Générez les getters/setters et le constructeur de cette classe.
 3. Écrivez la méthode **trouverParNom(String value)** qui permet de vérifier s'il existe une personne dans **personnes** avec un nom égal au paramètre **value**. Si c'est le cas, elle retourne le premier objet correspondant, sinon **null**.
 4. Écrivez la méthode **existeCodePostal(String value)** qui retourne **true** s'il existe une personne dans **personnes** qui a au moins une adresse dont le code postal égal au paramètre **value**, **false** sinon.
 5. Écrivez la méthode **compterPersonneVille(String value)** qui permet de calculer le nombre d'objets dans le tableau **personnes** ayant au moins une adresse dont l'attribut **ville** contient une valeur égale à celle du paramètre **value**.
 6. Écrivez la méthode **editerPersonneNom(String oldNom, String newNom)** qui remplace le nom des personnes ayant un nom égal à la valeur **oldNom** par celle de **newNom**
 7. Écrivez la méthode **editerPersonneCodePostal(String oldCP, String newCP)** qui remplace les codes postaux ayant une valeur égale à celle du paramètre **oldCP** par celle de **newCP**
 8. Dans la méthode **main** de la classe principale **Main**, testez toutes les méthodes réalisées dans les questions précédentes.

Exercice 4

Considérons une classe Java appelée **MaDate** ayant les trois attributs suivants :

- **jour** : un attribut privé de type entier.
 - **mois** : un attribut privé de type entier.
 - **année** : un attribut privé de type entier.
1. Créez la classe **MaDate**
 2. Redéfinissez la méthode **toString()** afin que nous puissions afficher une date sous le format **jour/mois/année**.
 3. Générez (ou écrivez) les getters et setters des trois attributs.

4. Définissez un constructeur avec trois paramètres `MaDate(int jour, int mois, int année)`
5. Écrivez la méthode `ajouterUnJour` qui permet d'ajouter un jour à notre date et faire des modifications, si nécessaire, pour les deux attributs `mois` et `année`. **Attention**, il faut traiter tous les cas. Par exemple si les trois attributs `jour`, `mois` et `année` contiennent respectivement les valeurs 31, 12 et 2016, alors la méthode `ajouterUnJour` doit affecter la valeur 1 à `jour`, 1 à `mois` et 2017 à `année`. Et s'ils contiennent 28, 02 et 2018 alors les nouvelles valeurs après modification seront respectivement 29, 02 et 2018.
6. Utilisez la méthode de la question précédente pour écrire la méthode `ajouterPlusieursJours(int n)` : `n` étant le nombre de jours à ajouter à la date enregistrée dans les trois attributs.
7. D'une façon similaire, définissez les méthodes `ajouterUnMois` et `ajouterUnAn()`.
8. Dans la méthode `main`, testez toutes les méthodes réalisées dans les questions précédentes.