Travaux pratiques N°2

Classes et Objets

Exercice 1 : Point

Considérons une classe Java appelée Point qui représente un point dans le plan en coordonnées cartésiennes. Cette classe devra avoir deux attributs privés de type double nommés x et y.

1. Créer la classe Point.
2. Générer les getters et setters pour les deux attributs.
3. Redéfinir la méthode toString() qui retourne la représentation mathématique d'un point : (x, y).
4. Définir un constructeur avec deux paramètres Point(double x, double y), et un constructeur sans paramètres Point() .
5. Ecrire 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.
   La distance entre deux points $A(x_1,y_1)$ et $B(x_1,y_1)$, en mathématiques, est égale à : $\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$
6. Ecrire la méthode 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, La méthode doit retourner un objet Point et pas les coordonnées.
   Les coordonnées d'un point $M(x_M ,y_M)$ milieu de $A(x_1 ,y_1)$ et $B(x_1 ,y_1)$, en mathématiques, sont :
   $$x_M = \frac{x_1 + x_2}{2},\ y_M = \frac{y_1 + y_2}{2}$$

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. Générer les getters et setters et le constructeur avec trois paramètres de la classe TroisPoints.
8. Ecrire une méthode sontAlignes() qui retourne true si les trois points premier, deuxième et troisième sont alignés, false sinon.
9. Ecrire une méthode estIsocele() qui retourne true si les trois points premier, deuxième et troisième forment un triangle isocèle, false sinon.
10. Ecrire une méthode estEquilateral() qui retourne true si les trois points premier, deuxième et troisième forment un triangle équilatéral, false sinon.
11. Ecrire une méthode estRectangle() qui retourne true si les trois points premier, deuxième et troisième forment un triangle rectangle, false sinon.
12. Dans la méthode main() de la classe principale Main, demandez à l'utilisateur de :
    - Saisir les coordonnées de trois points.
    - Afficher tous les détails sur ces trois points,
    - Afficher les milieux, les distances qui les séparent,
    - Afficher s'ils sont alignés, s'ils forment un triangle isocèle...

Exercice 2 : Nombre Complexe

1. Définir une classe Complexe, dont chaque instance représente un nombre complexe immuable de l'ensemble ℂ. Un objet complexe aura deux attributs, une partie réelle et une partie imaginaire : re+i.im.
2. La classe Complexe comportera un constructeur par défaut qui initialisera les deux attributs à zéro, ainsi qu'un constructeur qui initialisera un nombre complexe à partir de deux paramètres réels, et un troisième constructeur par recopie.
3. Générer une méthode *toString()*. La méthode *toString* permet la conversion d'un objet de type complexe en une chaîne de caractères.
4. Générer les getters.
5. Complétez la classe Complexe avec les opérations arithmétiques : *addition, soustraction, multiplication,* et *division*.
6. Ajoutez à la classe Complexe une méthode statique *Complexe additionner(Complexe a, Complexe b)*
7. Ecrire une méthode qui permet de calculer le module ainsi que l'argument d'un nombre complexe.
8. Ecrire une méthode qui permet de multiplier un nombre complexe par un scalaire.
9. Ecrire les méthodes Inverse, conjugué qui permettent de retourner l'inverse et le conjugué d'un nombre complexe.
10. Ecrire une méthode exponentielle qui calcule l'exponentielle d'un Complexe.
11. Redéfinir la méthode *equals()* pour comparer et vérifier si deux nombres complexes sont égaux ou non.
12. Ajoutez une méthode main faisant quelques tests élémentaires des complexes et leurs opérations.

En utilisant les principes de la POO résoudre une équation du second degré de la forme : $ax^2+ bx+ c=0$ dans ℂ. (a, b et c les coefficients du polynôme).

1. Définir une classe Equation.
2. Ecrire un constructeur à trois paramètres réels *a*, *b* et *c* qui correspondent aux coefficients du polynôme à résoudre. Ce constructeur affectera les valeurs passées en paramètre aux attributs *a*, *b* et *c* et calculera la valeur du discriminant delta.
3. Ecrire une méthode *afficherDiscriminant()* qui affiche la valeur du discriminant.
4. Ecrire une méthode *résoudre()* qui résout l'équation dans ℂ. (Utilisez la classe Complexe afin de représenter les racines de l'équation du second degré).
5. Ecrire une méthode *afficherSolutions()* qui affiche les solutions de l'équation.
6. Tester votre classe.

Exercice 3 : Stagiaire

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. Créer la classe Stagiaire
2. Générer les getters et setters des deux attributs.
3. Définir un constructeur avec deux paramètres Stagiaire(String nom, float[] notes)
4. Ecrire la méthode calculerMoyenne() qui permet de retourner la moyenne des notes d'un stagiaire
5. Ecrire les méthodes trouverMax() et trouverMin() qui permettent de retourner respectivement les notes max et min d'un stagiaire.
6. Considérons maintenant une classe appelée **Formation** ayant les attributs suivants :
   - intitule : un attribut privé de type chaîne de caractère
   - nbrJours : un attribut privé de type entier
   - stagiaires : un tableau d'objets de type Stagiaire
7. Créez la classe Formation, générez les getters et setters de ses attributs, et définissez le constructeur Formation(String intitule, int nbrJours, Stagiaire [] stagiaires)
8. Ecrire une méthode calculerMoyenneFormation() qui retourne la moyenne d'un objet de type formation (la moyenne des moyennes des stagiaires)
9. Ecrire une méthode getIndexMax() qui retourne l'indice du stagiaire dans le tableau stagiaires ayant la meilleure moyenne de la formation.
10. Ecrire une méthode nomMax() qui retourne le nom du premier stagiaire ayant la meilleure moyenne d'une formation.
11. Ecrire une méthode minMax() qui retourne la note minimale du premier stagiaire ayant la meilleure moyenne d'une formation.
12. Ecrire une méthode trouverMoyenneParNom(String nom) qui retourne la moyenne du premier stagiaire dont le nom est passé en paramètre.
13. 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 4 : Comptes Bancaires

L'objectif est de définir une classe permettant de modéliser des comptes bancaires. Cette classe Compte doit permettre à une application de créer et utiliser autant d'objet Compte que nécessaires.

- Un compte bancaire est identifié par un numéro de compte unique (entier positif attribué par la banque à l'ouverture du compte et ne peut être modifié par la suite). (N.B. Utiliser un compteur statique initialisé à 1000 pour attribuer automatiquement des numéros de compte : les comptes sont numérotés de 1001 à 1000+n, n étant le nombre de comptes qui ont été créés).
- Un compte est associé à une personne propriétaire du compte, cette personne est modéliser par une classe Client et décrite par son CIN (long), son nom (String) et son prénom (String). Une fois le compte créé, le propriétaire du compte ne peut plus être modifié. Un propriétaire peut avoir plusieurs comptes bancaires.
- La classe Client doit permettre d'afficher les informations d'un compte en fournissant le numéro de comptes et d'afficher les informations de l'ensemble des comptes du propriétaire.

- La somme d'argent disponible sur un compte est exprimée en Dirham (dh). Cette somme est désignée sous le terme de solde du compte. Ce solde est un nombre décimal qui peut être positif, nul ou éventuellement (et temporairement) négatif. Si le solde est négatif, on dit que le compte est à découvert. Le découvert d'un compte est nul si le solde du compte est positif ou nul, il est égal à la valeur absolue du solde si ce dernier est négatif.
- En aucun cas le solde d'un compte ne peut être inférieur à une valeur fixée pour ce compte. Cette valeur est définie comme étant - (moins) le découvert maximal autorisé pour ce compte. Par exemple pour un compte dont le découvert maximal autorisé est 2000 dhs, le solde ne pourra pas être inférieur à -2000 dhs. Le découvert maximal autorisé peut varier d'un compte à un autre, il est fixé arbitrairement par la banque à la création du compte et peut être ensuite révisé selon les modifications des revenus du propriétaire du compte.
- Créditer un compte consiste à ajouter un montant positif au solde du compte et débiter un compte consiste à retirer un montant positif au solde du compte. Le solde résultant ne doit en aucun cas être inférieur au découvert maximal autorisé pour ce compte.
- Lors d'une opération de retrait, un compte ne peut être débité d'un montant supérieur à une valeur désignée sous le terme de débit maximal autorisé. Comme le découvert maximal autorisé, le débit maximal autorisé peut varier d'un compte à un autre et est fixé arbitrairement par la banque à la création du compte. Il peut être ensuite révisé selon les modifications des revenus du propriétaire du compte.
- Effectuer un virement consiste à débiter un compte au profit d'un autre compte qui sera crédité du montant du débit.
- Lors de la création d'un compte le propriétaire du compte est indispensable. En l'absence de dépôt initial le solde est fixé à 0. Les valeurs par défaut pour le découvert maximal autorisé et le débit maximal autorisé sont respectivement de 800 dhs et 1000 dhs. Il est éventuellement possible d'attribuer d'autres valeurs à ces caractéristiques du compte lors de sa création.
- Toutes les informations concernant un compte peuvent être consultées : numéro du compte, nom et prénom du propriétaire, montant du découvert maximal autorisé, montant du débit maximal autorisé, situation du compte (est-il à découvert ou pas ?), montant du débit autorisé.

Travail demandé :

A partir du "cahier des charges" précédent élaborer une spécification de classes Java modélisant un propriétaire et un compte bancaire.

1. Il s'agira en analysant le texte ci-dessus de :
    - définir les attributs (variables d'instance, variables de classe) des deux classes,
    - d'identifier les méthodes publiques proposées par les deux classes,
    - de proposer un ou plusieurs constructeurs pour les deux classes.
2. Réaliser une implémentation en langage Java des deux classes précédemment spécifiées.
3. Ecrire un programme de test permettant de tester les deux classes.