Université de Yaoundé I
Département d'Informatique

Licence 3

INF3055 : Conception Orientée Objet

Année Académique 2021-2022

Fiche de TD N°1 : Principes de base de l'orienté objet

Valéry Monthé

NB :

1. *les codes sont à écrire en Java*
2. *vous devez à chaque fois réaliser une modélisation UML avant de l'implémenter*

Exercice 1 : Classe simple

On considère une classe ***Point*** pour manipuler les points du plan. Un point a des coordonnées (abscisse, ordonnée). Cette classe doit disposer des trois méthodes suivantes :

• ***initialiser***: *pour attribuer des valeurs aux coordonnées d'un point;*

• ***deplacer***: *pour modifier les coordonnées d'un point;*

• ***afficher***: *pour afficher un point ; par souci de simplicité, nous nous contenterons ici d'afficher* les coordonnées du point (cette méthode affichera : «je suis un point de coordonnées X et Y»).

1. Donner la modélisation UML de la classe Point ainsi définie.
2. Donner un squelette de code de cette classe en Java.

Exercice 2 : Héritage

1. Créer une classe ***Pointcol*** pour manipuler les points colorés du plan. Elle doit avoir un attribut ***couleur*** de type byte et une opération ***colorer()***
2. Modifier la classe Pointcol, en y ajoutant une méthode ***afficheCol***, qui en plus des coordonnées du point coloré, affiche sa couleur.
3. Modifier la classe ***Point***, y ajouter un constructeur. Puis ajouter également un constructeur dans la classe ***Pointcol,*** qui permet de construire les objets de type point coloré, directement avec leur couleur.

Exercice 3 : Polymorphisme

1. Modifier la classe ***Pointcol***, pour que sa méthode ***afficheCol*** s'appelle ***affiche***.
2. Modifier Les classes ***Point*** et ***Pointcol***, pour ne garder la méthode ***affiche*** que dans la classe ***Point***.
3. Définir une méthode ***identifie*** () qui affiche pour un :
    - Point : « **Je suis un point** »
    - Pointcol : « **Je suis un point coloré de couleur « couleur »** »
4. Ecrire un exemple de programme qui exploite les possibilités de polymorphisme pour créer un tableau "hétérogène" d'objets, c'est-à-dire dans lequel les éléments peuvent être de type différent (***Point***, ***Pointcol***). Parcourir le tableau et l'afficher.

Exercice 4 : Classe abstraite

1. Créer une classe abstraite ***Affichable,*** ayant juste la signature d'une méthode ***affiche***().
2. Dériver deux classes, qui construisent respectivement les entiers et les flottants et les affichent :
    i. Je suis un entier de valeur 25
    ii. Je suis un flottant de valeur 1.25

    NB : Utiliser un tableau d'objets de type différents

Exercice 5 : Interface

1. Transformer la classe abstraite ***Affichable,*** de l'exercice précédent en interface.
2. Modifier les classes **Entier** et **Flottant**, pour obtenir le même résultat qu'à l'exercice précédent.

Exercice 6 : Interface

Soit la classe Compte suivante

Compte
Numero : Chaine Proprietaire : Chaine Solde : Entier
Depot (montant : entier) Retrait (montant : entier)

1. Ecrire le code de l'implémentation en java de cette classe
2. Créer une classe **compte d'épargne** qui permet d'augmenter le solde suivant un certain taux.
3. On veut sécuriser la méthode retrait de manière à n'autoriser un retrait uniquement que si le solde est suffisant. Créer une classe **compte sécurisé** qui le fait.