TD4 POO / Java : les tableaux, composition et agrégation, polymorphisme

Objectif : Manipuler les tableaux en Java. Mettre en œuvre les situations de composition et d'agrégation.

Matériel / Logiciel : Environnement de développement sur PC (sous Linux) : JDK1.8 (ou supérieur) : compilateur java (javac) et JVM (java) / Eclipse.

Acquisition : tableaux de type de base et tableau de références (tableau d'objets).

1 Tableau de type de base

Réaliser une classe « Lucas » permettant de stocker les nombres de Lucas jusqu'à un indice max (20 par défaut). Les Nombres de Lucas sont définis par une suite mathématique utilisant la relation de récurrence suivante :

$$\begin{cases} U_0 = 2 \\ U_1 = 1 \\ U_{n+2} = U_{n+1} + U_n \end{cases}$$

Une méthode permet de récupérer le nombre de Lucas à partir de son indice (ex. $U_{10} = 123$) sans le recalculer à chaque appel.

Définir le modèle UML d'une telle classe.

Écrire les classes « Lucas » et « TestLucas ».

2 Classe Segment et tableau : composition

Modifier l'implémentation de la classe Segment du TD précédent en remplaçant les 2 attributs de type « Point » par un petit tableau (de 2 éléments).

Vous vérifierez que cette modification de la définition interne de cette classe est sans effet sur l'interface externe qu'elle propose : liste et signature des méthodes publiques disponibles.

3 Tableau de références (tableaux d'objets) : agrégation

On souhaite concevoir puis implémenter la classe «Ensemble» vue en CM, contenant des points (ou des points colorés). On pourra translater l'ensemble (vous réutiliserez pour cela la méthode « deplacer() » de la classe Point).

Définir le diagramme de classe UML de ce problème.

Écrire les classes «Ensemble» et « TestEnsemble ».

Vérifier le mécanisme du polymorphisme dynamique lors de l'affichage de l'ensemble contenant des points et des points colorés.

4 La classe Repertoire

On souhaite réaliser un petit répertoire téléphonique du même ordre que ceux qui existent sur les téléphones portables. On se limitera dans un premier temps aux informations indispensables à stocker (nom, téléphone et adresse mail / courriel, par exemple) pour chaque contact (comme définie lors du TD précédant) et aux fonctionnalités de base d'un répertoire, en particulier les possibilités d'ajout et de recherche (par nom) ou de recherche inverse (par téléphone) d'un contact dans le répertoire devront être implémentées. On ne prendra pas en compte dans cet exemple les situations éventuelles de doublons.

Définir le diagramme de classe UML de ce problème.

Écrire les classes « Repertoire » et « TestRepertoire ».