TD5 POO / Java : classes abstraites, interfaces, polymorphisme

Objectif : Manipuler les classes abstraites, les interfaces et le polymorphisme en Java.

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

Acquisition : classes abstraites, interfaces, polymorphisme.

1 Classe abstraite « AbstractFigure » (exemple du CM)

Réaliser une classe abstraite « AbstractFigure » telle que celle définie en CM permettant de mutualiser les éléments communs à toute figure géométrique (disque, carré, etc.). Vous prévoirez en particulier les 2 méthodes abstraites « calculerAire() » et « calculerPerimetre() ».

Définir le modèle UML de cet ensemble de classes.

Ajouter la classe « Rectangle » (dérivée elle aussi de « AbstractFigure ») mais plus générique que la classe de spécialisation « Carre ».

Écrire et tester les classes AbstractFigure, Disque, Rectangle, Carre.

2 Interface « Comptage »

Définir une interface « Comptage », possédant les 2 méthodes fonctionnelle « incrementer() » et « decrementert() » communes aux différents compteurs.

Définir le diagramme de classe UML de ce problème.

Écrire l'interface « Comptage » et modifier les classes Compteur, CompteurBorne, CompteurCyclique et Chronometre déjà réalisées, puis Tester.

3 Polymorphisme dynamique (les tests)

Reprendre les différents exemples et mettre en œuvre le polymorphisme dynamique dans les situations suivantes :

- les compteurs : définir une unique classe de test pour les 3 compteurs en utilisant systématiquement la classe Compteur lors de la déclaration des variables (objets de type Compteur, CompteurBorne, CompteurCyclique).
- Recommencer avec l'interface Comptage.
- Utiliser l'interface Comptage pour le test du chronomètre.
- Ajouter des points et des points colorés au plan (cf. TD précédent) et vérifier le mécanisme du polymorphisme dynamique lors de l'affichage du plan contenant des points et des points colorés.

4 Polymorphisme dynamique : la classe Dessin

On considère une classe Dessin composée de plusieurs figures géométriques. Les figures pourront être des disques, des carrés ou des rectangles. On ne prendra pas en compte le cas de figures qui se superposent. Une des méthodes de la classe « Dessin » retournera la somme des surfaces des différentes figures qui le composent.

Définir le diagramme de classe UML de ce problème.

Écrire les classes « Dessin « et « TestDessin ».

5 Interface et polymorphisme

Définir une interface Java pour le problème du répertoire téléphonique et tester.