TP_TD N° 2 : Programmation Orientée Objet avec JAVA

Exercice 1 :

a. Ecrire un programme qui demande à l'utilisateur au premier coup de saisir un entier, puis au deuxième coup une chaîne de caractère. Ensuite, il lui affiche l'entier et la chaîne de caractères qu'il a saisi. (Quel est le problème rencontré ? et comment le résoudre ?)

b. Modifier le programme précédent de telle façon qu'il demande cette fois à l'utilisateur au premier coup de saisir une chaîne de caractères, puis au deuxième coup une autre chaîne de caractère. Ensuite, il compare les deux chaînes de caractères et affiche « equal » si les deux chaînes sont les mêmes et « not equal » dans le cas contraire.

Exercice 2 :

1. Créer un package nommé Axe3D.
2. Dedans ce package, réaliser une classe *Point3D* permettant de représenter un point sur un axe 3D. Chaque point sera caractérisé par un nom (de type *char*) et ses coordonnés x, y et z (de type *double*) qui seront déclarés sans spécifier explicitement le type de modificateur. On prévoira :
   - un constructeur (public) recevant en arguments le nom et les coordonnées d'un point,
   - une méthode *affiche* sans modificateur d'accès imprimant (en fenêtre console) le nom du point et ses coordonnés.
   - une méthode *translate* sans modificateur d'accès effectuant une translation définie par les valeurs de ses arguments.
3. Dedans le même package, écrire un petit programme utilisant cette classe pour créer un point, en afficher les caractéristiques, le déplacer et en afficher à nouveau les caractéristiques.

Exercice 3 :

1. Reprendre le même exercice précédent, puis apporter les modifications suivantes sur la classe Point3D :
   - Supprimer la méthode *affiche*.
   - Supprimer la méthode *translate*.
   - Ajouter une méthode *toString* dont la signature est *public String toString()*, afin d'afficher les caractéristiques d'un point.
   - Ajoute une méthode *equals* pour comparer deux points. Cette méthode doit respecter la signature suivante : *public boolean equals(Point3D p)*

2. Cette fois après avoir supprimé la méthode translate de la classe Point3D, on vous demande de la déclarer et de l'implémenter dans la classe exécutable (programme principale) tout en respectant sa signature comme indiqué ci-dessous:
   public static void translate(Point3D p, double dx, double dy, double dz)
3. Modifier le petit programme utilisant cette classe pour créer trois points, puis afficher leurs caractéristiques à l'aide de la méthode toString, les déplacer à l'aide de la méthode translate, afficher à nouveau leurs caractéristiques et à la fin les comparer à l'aide de la méthode *equals*.

Exercice 4 :

Afin de respecter le principe de l'encapsulation, reprendre le même exercice précédent, puis effectuer la modification suivante :

- Mettez les différents attributs de la classe Point3D en mode private.

1. Essayer d'exécuter le programme principal à nouveau. Qu'est que vous constatez ?
2. Que doit-on faire pour résoudre le problème rencontré dans la question précédente ?
3. Effectuer les modifications nécessaires au niveau de la classe Point3D et le programme principale pour pouvoir l'exécuter à nouveau.

Exercice 5 :

Ecrire un programme utilisant une classe nommer Objet, destiné à contenir, à tout instant le nombre de type Objet déjà crées. Sa valeur est incrémentée de 1 à chaque appel du constructeur. Afficher ce nombre à chaque création d'un nouvel objet.

Exercice 6 :

On souhaite définir une classe Complexe pour manipuler facilement des nombres complexes.

1. Créer un nouveau package nommé NbComplexes
2. Dedans ce package, définir une classe Complexe comprenant deux champs de type double, qui représentent respectivement la partie réelle et la partie imaginaire d'un nombre complexe.
3. Définir le constructeur dont la signature est *public Complexe(double r, double i).*
4. Ecrire la méthode toString, dont la signature est *public String toString()* qui affiche un nombre complexe sous la forme : a+b*i ou a-b*i.
5. Créer un nouveau fichier Test.java. Tester le constructeur et la méthode d'affichage depuis la méthode main de ce fichier, en construisant différents nombres complexes.
6. Définir des méthodes pour l'addition et la multiplication de deux nombres complexes. Ces deux méthodes doivent prendre en argument un Complexe et doivent retourner un nouveau Complexe qui est le résultat de l'opération sur le nombre argument et le nombre sur lequel est appelée l'opération.

```
public Complexe additionner(Complexe c) {
…….
}

public Complexe multiplier(Complexe c) {
```

……

}

7. Tester depuis la méthode main du fichier Test.java l'addition et la multiplication de deux nombres complexes.
8. Comment tester l'égalité entre deux complexes ? Ecrire la méthode appropriée puis essayer de la tester depuis la méthode main du fichier Test.java.