Référence, objet, sous-typage, accesseur, égalité

Exercice 1 - Point

On cherche à écrire une classe `Point` stockant un point graphique en coordonnées cartésiennes (appelons les x et y).

1 Dans un premier temps, n'écrivez pas de constructeur mais seulement une méthode `String toString()` permettant d'afficher sous forme textuelle les coordonnées d'un point.
Puis essayer le code suivant :

```
Point p=new Point();
System.out.println(p);
```

Expliquer.
Note: il est possible d'utiliser la commande `javap` pour voir ce que le compilateur génère comme `byte-code`.

2 Ajouter un constructeur initialisant les coordonés du point avec deux paramètres. Essayer de réexecuter le code ci-dessus. Expliquer.

3 Écrire deux accesseurs permettant d'obtenir les valeurs des coordonnées (`getX()` et `getY()`). Quel est l'intérêt d'utiliser des accesseurs plutôt que de mettre les champs publics ?

4 Pourquoi est-ce qu'il ne faut pas obligatoirement mettre des accesseurs lorsque l'on crée une classe ?

5 Écrire un autre constructeur permettant d'initialiser un point sans passer de paramètre. Le point aura pour coordonnées (0,0). Écrire un troisième constructeur qui prend un point en paramètre et et utilise les coordonnées de celui-ci pour s'initialiser.

6 Écrire une méthode `translate(int dx,int dy)` qui permette de translater un point d'une valeur de décalage en x et y.

7 Que fait le programme suivant ?

```
Point p=new Point(2,3);
Point p2=p;

p.translate(1,1);
System.out.println(p);
System.out.println(p2);
```

Expliquer.

Exercice 2 - Test d'égalité

En utilisant la classe `Point` de l'exercice précédent. Qu'affiche le code ci-dessous :

```
Point p=new Point(1,2);
Point p2=p;
Point p3=new Point(1,2);

System.out.println(p1==p2);
System.out.println(p1==p3);
```

1. Écrire dans la classe `Point` une méthode `isSameAs()` (à vous de trouver la signature exacte de la méthode) qui renvoie `true` si deux points ont les mêmes coordonnées.
2. Remarquons qu'il existe déjà une méthode nommé `equals()` dans la classe `Object` dont le rôle est de tester si deux objets sont égaux sémantiquement.
   Transformer la méthode `isSameAs()` en méthode `equals()`.
3. Qu'affiche le code suivant :

```
Object p=new Point(1,2);
Object p2=new Point(1,2);

System.out.println(p.equals(p2));
```

   Expliquer ce résultat. Comment obtenir un résultat plus logique ?
4. Dans quel cas utilise t'on l'annotation `@Override` ?

Exercice 3 - Circle

Écrire une classe `Circle`, un cercle étant définie par un point correspondant au centre et un rayon.

1. Écrire le constructeur du `Circle`.
2. Écrire la méthode `toString` qui affiche le centre et la rayon.
3. Écrire la méthode `translate(int dx,int dy)` qui translate le cercle. Qu'affiche le code suivant :

```
Point p=new Point(1,2);
Circle c=new Circle(p,1);

Circle c2=new Circle(p,2);
c2.translate(1,1);

System.out.println(c+' '+c2);
```

   Expliquer.
4. Écrire la méthode `equals()` qui renvoie vrai si deux cercles ont le même centre et le même rayon.
5. Écrire la méthode `contains()` qui renvoie vrai si un point est contenu dans un cercle.
6. Écrire la méthode `contains(Point p,Circle... circles)` qui renvoie vrai si un point est contenu dans un des cercles.