

Première partie

L'objectif de cette première partie de TP est de vous initier à **Java** et de vous donner l'occasion d'écrire quelques classes après avoir compris comment étaient programmées d'autres classes qui sont fournies.

Nous souhaitons définir le contexte informatique nécessaire à la représentation d'objets graphiques variés. Pour ce faire, nous allons d'abord décrire les objets les plus simples (à commencer par le **Point**) pour **ensuite** nous pencher sur des objets plus complexes (**Segment** puis des *polygones*, des *rectangles*, etc.). Nous nous attacherons alors à afficher graphiquement des objets des classes ainsi introduites.

Pour cette partie, on fournit les classes dans les fichiers suivants :

- `Point.java` (à compléter)
- `Segment0.java` (presque complète)
- `Test0.java` (pour vérifier l'exécution)

Rappel : la documentation Java est disponible à l'adresse suivante : <http://docs.oracle.com/javase/8/docs/api/index.html>

1 Le point

Nous cherchons ici à construire une classe **Point** représentant un point de l'espace cartésien à deux dimensions à coordonnées réelles. Nous utiliserons par la suite les points pour définir des figures géométriques plus complexe.

Question 1.

- **Définir** une classe définissant un *point* (en deux dimensions, avec des coordonnées entières) et permettant au moins les opérations suivantes :
 - créer un point à une position prédéfinie ;
 - créer un point à partir d'un point déjà existant (*i.e.* copier un point existant) ;
 - accéder à l'abscisse du point ;
 - accéder à l'ordonnée du point ;
 - traduire le point.
 - afficher les coordonnées du point (cette méthode est fournie).
- **Programmer** en Java en complétant : la classe **Point** (fournie) pour permettre que la classe **Test0** (fournie) donne un résultat.

*Cette classe servira ensuite pour obtenir le résultat de l'image ci-dessous en utilisant les classes **Segment** et **SegmentColore** (voir question 2 ci-dessous) puis les figures géométriques de la deuxième partie.*
- **Exécuter** le programme principal dans la classe **Test0** et observer le résultat obtenu.

2 Les segments

Question 2.

- **Comprendre** le programme Java de la classe **Segment0** qui a les méthodes suivantes pour :
 - créer un segment vide (donc initialement sans sommet) ;
 - créer un segment à partir d'un segment déjà existant (*i.e.* copier un segment existant) ;
 - traduire le segment ;
 - plus quelques "accesseurs" (qui serviront pour la suite) ...
- **Programmer une nouvelle méthode** : `ajouterSommet(...)` dans **Segment0**. Celle-ci permet de fixer les deux extrémités d'un segment qui avait été défini comme vide. Elle est utilisée dans **Test0**.
- **Tester** votre classe **Segment0** avec sa nouvelle méthode en exécutant le programme dans **Test0**.

*Indication : les différents points définissant le segment (c'est-à-dire ses extrémités) sont stockés dans une structure : `Vector<T>` (*T* étant le type générique objet des éléments constitutifs de la liste, donc ici des **Point**).*

Consulter la documentation Java pour connaître les différentes méthodes qui s'appliquent sur cette classe (`add()`, `get()`, `size()`, ...).

Deuxième partie

L'objectif de cette seconde partie de TP est de visualiser les objets graphiques qui ont été définis dans la précédente partie et d'en créer de nouveaux. Pour ce faire, les classes suivantes sont fournies :

- une classe `ObjetGraphique` définissant un objet graphique;
- une classe `Affichage` permettant de construire une fenêtre, de l'afficher à l'écran et de mettre à l'intérieur des objets graphiques qui s'afficheront.
- une nouvelle classe `Segment` (un peu plus complète que `Segment0` pour pouvoir être un objet graphique);
- une nouvelle classe `Test` qui permettra des tester les nouvelles classes à programmer dans cette partie;

3 Les polygones

Question 3.0 • **Utiliser** la classe `Test` pour afficher des segments comme sur la figure ci-dessous.

Par la suite, ce programme de test consistera à créer des polygones (colorés ou non), à les stocker dans le vecteur d'`ObjetGraphique` pour les rendre affichable.

Question 3.1 • **Créer** une classe `Polygone` permettant au moins les opérations suivantes :

- créer un polygone;
 - tradater le polygone;
 - transformer le polygone pour lui ajouter un nouveau sommet.
- **Montrer** le bon fonctionnement de cette nouvelle classe en modifiant `Test`. **Afficher** quelques polygones.

Question 3.2 • **Créer** une classe `PolygoneColoré` permettant au moins les opérations suivantes :

- créer un polygone coloré;
 - créer un polygone coloré à partir d'un polygone déjà existant (i.e. copier un polygone existant) en le (re-)colorant;
 - tradater le polygone coloré;
 - ajouter un nouveau sommet.
- **Montrer** le bon fonctionnement de la classe ainsi définie. **Afficher** quelques polygones colorés.

4 Héritage de la classe `Polygone` pour construire des Rectangles

Question 5. [question bonus] Définir et implanter en Java une classe `RectangleColoré` permettant au moins les opérations suivantes :

- créer un rectangle coloré;
- créer un rectangle coloré à partir d'un rectangle déjà existant;
- tradater le rectangle coloré.

Montrer le bon fonctionnement de la classe ainsi définie. **Afficher** quelques rectangles colorés.

