I.U.T. de NICE / COTE D'AZUR | Année 2015_2016
Département Informatique | Module M2103
DUT S2A | **Semaine_2**

- -

L'objectif des exercices ci-dessous est double :

a) mettre en œuvre **l'IDE JCreator** couplé au **JDK 1.7** (si vous ne l'avez pas déjà fait),

b) développer une classe **Segment** qui met en œuvre la relation de composition.

Ces travaux doivent être exécutés sur les postes de travail du Département Informatique. L'IDE est pré installé dans l'espace des logiciels de développement.

Exercice 1. Configuration de l'IDE JCreator

A l'aide du tutorial en ligne de l'IDE, configurer ce dernier pour le coupler avec le JDK de référence. Par défaut, l'IDE n'a pas de lien pré établi avec un JDK.
Compiler et exécuter avec succès le point d'entrée de la classe **T_Point**.

Exercice 2. Finalisation de la feuille d'exercices de la semaine 1

Terminer les exercices de la semaine précédente en utilisant l'IDE.

Exercice 3. Création de l'arborescence de TD&TP pour la séance 2

Dans votre répertoire de P : dédié au module M213, créer un sous répertoire propre à la séance 2, de façon identique à ce que vous fait pour la première séance.
Copier dans le sous répertoire Exercice_4, les fichiers sources solutions des exercices 11 et 12 de la semaine précédente.

Exercice 4. Ajout de la méthode confondus dans la classe Point

Développer (dans Point.java) et tester (dans T_Point.java) la méthode suivante qui retourne true si le point support et le point passé en paramètre sont confondus, faux sinon :

```
public boolean confondus (Point op2)
```

On dit que deux points A (xA, yA) et B (xB, yB) sont confondus si leurs coordonnées respectivement sur Ox et sur Oy sont « proches » :

$$|xA - xB| \leq \varepsilon \ et \ |yA - yB| \leq \varepsilon$$

On utilisera la constante EPSILON= 1./1000.

Exercice 5. Classe Segment / Relation de composition

Détailler en langage Java la classe **Segment** présentée en cours : un segment est <u>**composé**</u> de 2 points <u>**distincts**</u>, que l'on nommera origine et extrémité.

On se limitera dans un premier temps les codes sources à la déclaration des attributs, la définition des constructeurs et des accesseurs en lecture et de la méthode ***toString***.
Compiler avec succès cette nouvelle classe.

Exercice 6. Test de la classe Segment

Par analogie avec la classe **T_Point**, détailler en langage Java un programme de test de la classe **Segment** (Classe **T_Segment**).
Compiler et exécuter ce dernier avec succès.

Exercice 7. Calcul de la longueur d'un segment

Introduire dans la classe **Segment** une méthode ***longueur*** qui retournera la longueur de l'objet support.
Compiler avec succès la nouvelle version de cette classe.
Modifier le programme de test **T_Segment** pour y introduire des exemples d'appel de la nouvelle méthode et exécuter ce dernier avec succès.

Exercice 8. Autres modifications de la classe Segment

Introduire dans la classe **Segment** deux nouvelles méthodes ***projX*** et ***projY*** permettant respectivement de calculer le projeté d'un segment support sur l'axe des abscisses et sur l'axe des ordonnées.
Compiler avec succès cette nouvelle version de cette classe.
Modifier le programme de test de la classe **Segment** pour y introduire des exemples d'appel des deux nouvelles méthodes et exécuter ce dernier avec succès.

Exercice 9. Modélisation des transformations du plan

La modélisation du plan euclidien définit les transformations standards suivantes sur un segment : translation, rotation, symétrie centrale et homothétie.
Introduire les méthodes correspondantes dans la classe **Segment**.
Compiler avec succès la nouvelle version de cette classe.

Exercice 10. Nouvelle mise en œuvre de la classe Segment

Modifier le programme de mise en œuvre de la classe **Segment** pour y introduire des exemples d'appel des nouvelles méthodes et exécuter ce dernier avec succès.