- -

L'objectif des exercices ci-dessous est de mettre en œuvre le **JDK 1.7** par **lignes de commandes** et de créer les premières classes Java.

Ces travaux doivent être exécutés sur les postes de travail du Département Informatique. Le JDK support est pré installé dans l’espace des logiciels de développement.

Exercice 1. Chaîne de production de programmes en langage C (révision)

Enumérer les éléments de la chaîne de production de programmes exploitée dans les modules M112 et M113.

Détailler le rôle précis de chacun d’eux, le type et l’extension des fichiers (flux) produits en sortie de chacun d’eux.

Exercice 2. Identification des exécutables du JDK

Identifier les exécutables suivants dans le sous répertoire ***bin*** du JDK :

- Compilateur
- Machine virtuelle Java (JVM)
- Outil de production de la documentation de programmation
- Outil de production des fichiers d’archive

Détailler le rôle précis de chacun d’eux, le type et l’extension des fichiers (flux) produits en sortie de chacun d’eux. Pour mener à bien ce travail, il est recommandé de s’appuyer sur la documentation en ligne Oracle et le fichier readme du JDK.

Exercice 3. Création d’une arborescence de TD&TP

Sur l’unité logique P :, créer une arborescence physique de sous répertoires dédiée au module M2103.

Cette arborescence devra permettre de localiser de **façon aisée** : l’énoncé, les annexes et tous les fichiers source qui seront manipulés pour chaque exercice de chaque séance de TD/TP.

Recopier dans cette arborescence l’énoncé et toutes les annexes de la présente feuille d’exercice.

Cette action devra être reproduite ensuite pour chaque séance de TD/TP.

Exercice 4. Options de compilation Java

Dans une "invite de commandes" lancée au préalable, exécuter le compilateur Java, sans aucun argument.

Q1 Quel est le rôle de l’option ***–classpath*** ?
Q2 Quel est le rôle de l’option ***–version*** ?
Q3 Quel est le rôle de l’option ***–d*** ?

Exercice 5. Compilation d'une première classe

Dans une "invite de commandes" lancée au préalable, compiler avec succès la classe **Point** fournie en annexe. Identifier le nom du fichier "bytes code" produit par le compilateur.

Modifier le nom du fichier source et vérifier les contraintes de nommage imposées par le langage.

Exercice 6. Exécution en ligne d'un premier programme

Analyser avec soin le programme source fourni en annexe.

Compiler avec succès et exécuter ce programme.

Exercice 7. Documentation de programmation

Enumérer et identifier le rôle précis des balises de documentation exploitées dans le code source de la classe **Point**.

Produire et analyser avec soin la documentation de programmation de cette classe.

Exercice 8. Modifications de la classe Point

Introduire dans la classe **Point** deux nouvelles méthodes ***projX*** et ***projY*** permettant respectivement de calculer le projeté d'un point support sur l'axe des abscisses et sur l'axe des ordonnées.

Compiler avec succès cette nouvelle version.

Exercice 9. Modifications de la mise en œuvre de la classe Point

Mettre en œuvre les deux nouvelles méthodes dans le programme existant des exemples de la classe **Point**.

Compiler avec succès et exécuter ce programme.

Exercice 10. Analyse et compilation de la classe Vecteur

Analyser avec soin les codes sources de la classe **Vecteur** fournie en annexe.

Dans une "invite de commandes" lancée au préalable, compiler avec succès cette classe **Vecteur**.

Exercice 11. Exécution en ligne d'un second programme

Analyser avec soin le programme source fourni en annexe.

Compiler avec succès et exécuter ce programme.

Exercice 12. Modélisation des transformations du plan

La modélisation du plan euclidien définit les transformations standards suivantes : translation, rotation, symétrie centrale et homothétie. Certaines d'entre elles exploitent des vecteurs.

Introduire les méthodes correspondantes dans la classe **Point**.

Compiler avec succès la nouvelle version de cette classe.

Exercice 13. Nouvelle mise en œuvre de la classe Point

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