# Travaux Dirigés de Java n°5

*—Maîtrise d'informatique—*

▸ **Exercice 1. (Écriture fichier texte)** *Écrire une fonction réalisant l'écriture d'un tableau de* `String` *dans un fichier texte (lisible et modifiable avec un éditeur de texte). En déduire un programme réalisant la sauvegarde des arguments qu'on lui donne sur sa ligne de commande dans un fichier texte.*

▸ **Exercice 2. (Lecture fichier texte)** *Écrire une fonction permettant de récupérer les données écrites dans le fichier par la fonction précédente. Ecrire une fonction réalisant l'affichage des données récupérées, et l'utiliser dans un programme.*

▸ **Exercice 3. (Écriture/Lecture fichier binaire)** *Écrire une fonction réalisant l'écriture d'un tableau d'objets quelconques dans un fichier (cette fois-ci au format de java; on utilisera la propriété de* sérialisation *des objets). Écrire également la fonction de lecture. Utiliser les deux fonctions pour sauvegarder et lire les arguments qu'on donne au programme sur sa ligne de commande dans un fichier de nom* `toto`*. Les utiliser également pour sauver/lire une liste chaînée construite dans le dernier TD.*

▸ **Exercice 4. (Collections)** *On souhaite maintenant que nos fonctions de lecture mettent les données lues dans une structure de donnée triée (par ordre lexicographique pour les chaînes de caractères, par exemple). Changer les fonctions de lecture pour qu'elles prennent en plus en argument une telle structure de donnée, qu'elles auront à charge de remplir. Faire un programme qui réalise une lecture du fichier* `toto`*, et affiche à l'aide d'un itérateur la structure de donnée triée. Nous souhaitons maintenant changer l'ordre du tri (par exemple, ne plus trier sur l'ordre lexicographique mais sur la longueur des chaînes). Comment faire ? Écrire un programme qui réalise une lecture de* `toto`*, et affiche à l'aide d'un itérateur la structure de donnée triée suivant le nouvel ordre.*

▸ **Exercice 5.** *Les fonctions de l'exercice précédent ont deux gros défauts:*

- *le tri est réalisé au fur et à mesure des insertions dans la structure, ce qui nous donne une complexité en temps quadratique sur le nombre d'éléments de la structure,*
- *elles prennent en argument des* `Collection`*, or il existe en Java des structures de données destinées à recevoir des élémetns qui ne sont pas des* `Collection` *(par exemple, les* `Map`*.*

*Comment corriger ces deux défauts ?*