ESIAG - UPEC
Diplôme d'ingénieur Systèmes d'Information
Ing1 – Programmation pour la communication

Feuille d'exercices n°1

Exercice 1 : Écrire un programme qui lit un fichier texte et le modifie afin qu'il contienne le même texte mais avec un caractère « : » entre chaque caractère du texte. Trois solutions sont à expérimenter :

1. Lire tout le texte et faire un tableau de caractères, intercaler le caractère « : » entre chaque case du tableau, écrire le tableau modifier dans le fichier.
2. Ouvrir simultanément en lecture le fichier donné et en écriture un fichier temporaire, lire chaque caractère du fichier donné et écrire dans le fichier temporaire le caractère lu puis le caractère « : », supprimer le fichier d'origine et renommer le fichier temporaire pour lui donner le nom du fichier d'origine.
3. Utiliser la lecture/écriture bas niveau pour effectuer directement dans le fichier la mise à jour de la taille puis la modification du contenu.

Exercice 2 : Utiliser deux fichiers pour simuler un canal de communication entre deux programmes : chaque programme affichera dans le terminal ce qu'il lit en entrée et écrira en sortie ce que l'utilisateur saisit au clavier. Faire un mode « synchrone » : lecture et écriture strictement alternée. Faire un mode « asynchrone » : lecture et écriture imprévisibles.

Exercice 3 :

1. Écrire un programme qui recopie un fichier texte XXX.txt pour créer un autre fichier texte YYY.txt. Les noms des deux fichiers seront dans une première version saisis directement au clavier puis donnés sur la ligne de commande lors du lancement de l'application : java MonProgramme XXX.txt YYY.txt.
   Attention : si le fichier XXX.txt n'existe pas, le programme affichera un message d'erreur et s'arrêtera.
2. Modifier la première version du programme pour qu'il propose une boucle d'interaction qui permette à l'utilisateur de donner des commandes. Une commande sera donnée par un texte sur une seule ligne que le programme décodera après que l'utilisateur ait appuyé sur la touche « entrée ». Lorsque la commande est inconnue, le programme affichera « Commande inconnue » et reprendra la boucle pour attendre la commande suivante.
   Mettre en place une commande « cp » permettant de recopier un fichier de la même façon que dans le point 1. : cp XXX.txt YYY.txt effectuera la copie puis reprendra la boucle d'interaction.
   Attention : si le fichier XXX.txt n'existe pas, le programme affichera un message d'erreur et reprendra la boucle d'interaction.
3. Ajouter les commandes :
   - mv XXX.txt YYY.txt : qui transforme le nom du fichier.
   - rm XXX.txt : qui supprime le fichier.
   - pwd : qui affiche le répertoire courant.
   - cd XXX : qui modifie le répertoire courant pour entrer dans le sous répertoire XXX. Attention : si XXX vaut « .. » il faut alors remonter dans le répertoire parent ; si XXX vaut « . » il ne faut pas changer le répertoire courant. Attention : si XXX commence par un « / » il faut placer le répertoire courant sur le chemin absolu XXX.

- ls : qui affiche les entrées (fichiers et sous-répertoires) du répertoire courant.
- find XYZ : qui affiche toutes les entrées du répertoire courant qui contiennent dans leur nom la chaîne de caractère XYZ.
- findrec XYZ : qui affiche toutes les entrées du répertoire courant et des sous-répertoires qui contiennent dans leur nom la chaîne de caractère XYZ en préfixant systématiquement le noms de l'entrée par le chemin permettant d'y accéder (exemple : toto/titi.zut ou truc/machin/bidule.tralala ou ./zinzin si c'est dans le répertoire courant).
-

Attention : selon les commandes, si les fichiers ou chemins indiqués n'existent pas, afficher un message d'erreur et reprendre la boucle d'interaction.

Exercice 4 : On considère une application qui gère un carnet d'adresse. Chaque entrée est un contact possédant un nom, un prénom, une adresse (numéro dans la rue, nom de la rue, nom de la ville, code postal), un numéro de téléphone, une adresse mail. On souhaite pouvoir :

- créer un nouveau contact ;
- supprimer un contact existant ;
- modifier un contact existant ;
- afficher la liste des contacts ;
- afficher le détail d'un contact ;
- sauvegarder le carnet d'adresse dans un fichier dont le nom est indiqué par l'utilisateur ;
- restaurer le carnet d'adresse depuis un fichier dont le nom est indiqué par l'utilisateur.

Pour la sauvegarde/restauration, faire une version avec et une autre sans la serialization. Proposer une version en mode texte puis une autre avec une interface graphique.

Exercice 5 : On considère une application qui gère la scolarité d'un groupe d'étudiants. Un étudiant possède un nom, un prénom, un numéro, une formation et un ensemble de notes correspondant à sa formation. Une formation est un ensemble de matières, chaque matière ayant un nom et un coefficient. Afin de garantir l'anonymat, trois fichiers seront prévus pour gérer les données : un fichier de formations contenant toutes les informations s'y rapportant ; un fichier étudiant contenant les informations des étudiants sauf leurs notes ; un fichier de notes contenant les numéros des étudiants et leurs notes ainsi que leur moyenne calculée en fonction des coefficients.
Écrire une application en mode texte (interface graphique dans une version améliorée) permettant : de saisir la description des formations, des étudiants, les notes, et de sauver/charger les fichiers à tout moment sur les différents éléments. Une commande spécifique lancera le calcul des résultats et mettra à jour le fichier correspondant. Faire deux versions avec/sans la serialization.

Exercice 6 : Faire un mini-tableur avec une interface graphique permettant de saisir/visualiser les données du tableau, de les sauvegarder et de les charger. Ajouter la possibilité de saisir des formules de calcul prenant les données d'autres cellules.