TP4: les fichiers

Exercice 1

Écrire une fonction `identiques()` qui permet de tester si deux fichiers textes sont identiques ou non. Les noms des deux fichiers sont saisis au clavier.

Exercice 2

Écrire une fonction `fusion_fichiers()` qui permet de fusionner deux fichiers textes dont les noms sont saisis au clavier.

Exercice 3

On suppose avoir créé un fichier texte contenant une matrice. Chaque ligne du fichier une ligne de la matrice, et les éléments de chaque ligne sont séparés par le caractère espace.

$5,7$	$3,4$	$0,0$	$-0,2$	$-1,5$	$-5,0$
$6,9$	$1,0$	$2,0$	$1,0$	$2,0$	$2,0$
$-2,0$	$1,7$	$1,0$	$-1,0$	$2,0$	$2,0$
$-5,0$	$5,0$	$4,0$	$0,3$	$-1,0$	$-1,0$
$-3,3$	$1,0$	$5,0$	$-1,0$	$1,0$	$1,0$
$-1,0$	$2,0$	$1,0$	$5,0$	$0,1$	$3,0$

1. Écrire une fonction `ChargerMatrice(ch)` qui prend en paramètre une chaîne de caractères `ch` qui contient le chemin absolu du fichier texte contenant une matrice et qui retourne la matrice sous forme de liste des listes.
2. Écrire une fonction `TransposeMatrice(ch)` qui permet de calculer le transposé de la matrice et enregistre le résultat dans un autre fichier binaire `'transposeM.bin'`.

Exercice 4

Soit un fichier intitulé `'concours.dat'` qui comporte les enregistrements relatifs aux candidats d'un concours. Chaque enregistrement est composé de : ID, nom, prenom, age, notes, moyG, decision.

Le champ note se compose de trois notes et le champ decision prend les valeurs suivantes :

— **admis** si MoyG ≥ 12,

— **refusé** si $6 \leq$ MoyG < 12,

— **ajourné** sinon.

1. Définir la fonction `saisir_un_etudiant()` qui permet de remplir les données relatives à un seul candidat sous forme d'un dictionnaire et l'enregistrer dans le fichier `'concours.dat'`.
2. Écrire une fonction `saisirAll(N)` qui permet de remplir les données relatives à N candidats dans le fichier `'concours.dat'`.
3. Écrire une fonction `recherche_par_decision(dec)` qui permet de retourner une liste contenant les candidats ayant la décision `dec` s'il y en a dans le fichier `'concours.dat'`.
4. Définir la fonction `Admis()` qui permet de créer le fichier `'admis.txt'` comportant les données (nom, prenom et moyG) relatives aux candidats admis mais classés par ordre de mérite.
5. Afin de sélectionner en priorité les 10 premiers candidats admis, créer la fonction `Attente()` qui produira à partir du fichier `'admis.txt'`, deux nouveaux fichiers intitulés `'admis10.txt'` et `'attente.txt'` comportant les données relatives aux candidats admis mais dont le classement dépasse 10.