Examen de Fin de Semestre

Exercice 1 :

En mathématiques, deux nombres entiers sont dits amicaux si la somme des diviseurs de l'un est égale à la somme des diviseurs de l'autre et si ces deux sommes valent la somme des deux nombres. Par exemple 220 et 284 sont amicaux.

1. Ecrire une fonction qui vérifie si deux nombres sont amicaux.
2. Ecrire un programme qui permet de compacter un tableau de n valeurs entières en supprimant tous les nombres amicaux d'une valeur **val** donnée (des 0 seront insérés à la fin du tableau).

Exercice 2 :

L'**anagramme** est obtenue en inversant ou en permutant les lettres d'un mot ou d'un groupe de mots pour en extraire un sens ou un mot nouveau. Exemple *"aube"* et *"beau"*.

1. Ecrire une fonction qui vérifie si deux mots sont anagrammes.

Les anagrammes peuvent aussi être appliquées à plusieurs mots (phrases). Exemple *"les miserables"* et *"il les embrasse"*.

2. Ecrire une fonction qui vérifie si deux phrases sont anagrammes (utiliser la fonction de la question 1).
3. Soit une matrice A. Chaque ligne de la matrice contient un mot. Ecrire un programme qui affiche tous les mots de A qui sont anagrammes avec le premier mot de A.

Exercice 3 :

Nous nous intéressons ici à une représentation économique du point de vue de l'espace mémoire des matrices creuses, c'est-à-dire des matrices comportant « beaucoup » de zéros. L'idée est de maintenir de telles matrices en ne représentant que les coefficients non nuls.

Pour chaque élément non nul de la matrice, on crée un enregistrement de trois champs contenant la valeur, le numéro de colonne et un pointeur vers l'élément non nul suivant de sa ligne. On utilise de plus un tableau *ligne* qui contient les adresses du premier élément non nul de chaque ligne.

1. Ecrire les déclarations des structures de données d'une telle représentation.
2. Ecrire une fonction qui permet de représenter une matrice ordinaire par la nouvelle représentation.
3. Ecrire un programme qui permet de réaliser la somme de deux matrices ayant les mêmes dimensions dans la nouvelle représentation.
4. On veur créer une Pile qui sauvegarde pour chaque ligne : le numero de la ligne, le nombre d'éléments non nuls et l'adresse du premier élément de la ligne. Cette pile doit être triée selon le nombre décroissant des éléments non nuls des lignes.
   Ecrire le programme qui construit cette pile.