Licence 2 Informatique

Algorithmique et programmation 3
EXERCICES

3-Listes, piles, files

Exercice 3-1 : utilisation des primitives d'accès aux listes

Dans toutes ces questions, vous devez utiliser les primitives d'accès aux listes vues en cours et donner la complexité asymptotique dans le pire des cas de vos algos.

a. Ecrire un sous-programme itératif créant la liste de toutes les valeurs d'une autre liste qui ont la propriété d'être supérieures à la somme de toutes les valeurs précédentes.

   Exemple : si l'entrée est (8, -1, 9, 4, 21, -50, -7, 3)
   alors la sortie est (8, 9, 21, -7, 3) (ou n'importe quelle permutation de ces valeurs)

b. Ecrire une version itérative et une version récursive d'une fonction dont les données sont V et N et dont le résultat est une liste chaînée contenant N fois la valeur V.

c. Ecrire une procédure itérative puis récursive qui supprime toutes les valeurs négatives d'une liste chaînée.

Exercice 3-2 : toutes les permutations des N premiers entiers

Savoir réaliser toutes les permutations des N premiers entiers permet de trouver toutes les permutations d'un tableau de N valeurs : il suffit de considérer que chaque permutation des N premiers entiers (par exemple 3,4,1,2) dit dans quel ordre prendre les éléments du tableau (troisième, quatrième, premier et deuxième).

a. En utilisant les primitives d'accès aux listes chaînées, écrire une fonction récursive à trois paramètres : une liste d'entiers L, un entier Pos et un entier V. La fonction doit restituer la liste obtenue en insérant V à la position Pos de la liste L. *La liste restituée ne doit partager aucune mémoire avec L.*
   Si Pos vaut 1, il s'agit d'une insertion en tête, s'il vaut (N+1), d'une insertion en queue.

b. Ecrire un algorithme récursif de calcul de toutes les permutations des entiers de 1 à N. Chaque permutation est une liste d'entiers et la liste des permutations est une liste de listes d'entiers.
   Une difficulté de cet exercice vient de l'usage de listes de listes d'entiers.