Examen Structures de données
Lundi 30 janvier 2006

problème I
Structure de tableau

On considère un tableau T qui contient des nombres entiers relatifs (positifs, négatifs ou nuls) sans ordre particulier.

Question 1 *Ecrire un algorithme qui déplace les éléments du tableau (sans utiliser un autre tableau) de façon que tous les nombres négatifs se trouvent avant les nombres nuls ou positifs. Après l'exécution de l'algorithme, on doit pouvoir vérifier que :*

Pour tous i et j, tels que $i < j$, si $T[j] < 0$ alors $T[i] < 0$.

Une suggestion : partitionner le tableau par rapport 0 : supposons qu'entre les indices 1 et neg, tous les $T[i]$ sont négatifs et que, entre les indices pos et n, tous les $T[i]$ sont positifs. Alors on considère $T[neg + 1]$, s'il est négatif, on passe au suivant, s'il est positif on l'échange avec $T[pos]$.

Question 2 *Quelle est la complexité de votre algorithme ?*

problème II
Gestion d'un parc de voitures

Vous disposez d'un parc de 50 véhicules à la disposition des collaborateurs de votre entreprise qui ont à effectuer des déplacements à but professionnel. Vous êtes chargé de le gérer en tenant compte des besoins (dates, nb de places) ainsi que des périodes de révision des véhicules pendant lesquelles ceux-ci ne sont pas utilisables.

Question 1 *Proposer une stratégie de gestion du parc et une structure de données permettant de programmer cette stratégie.*

problème III
Tri par Tas

Soit T le tableau suivant :

3	5	1	6	8	11	2	4	9	10

Question 1 *Construire un tas min correspondant à l'ensemble de ces éléments. Représenter ce tous sous la forme d'un tableau et sous la forme d'un arbre.*

Question 2 *Effectuer un tri par tas à partir du tas obtenu à la question précédente. Pour chaque itération représenter le tableau résultant ainsi que le tas correspondant aux éléments non encore triés sous la forme d'un arbre.*

problème IV
Liste ordonnée

Soit L une liste simplement chaînée dans laquelle chaque élément est un entier. L est une liste ordonnée si pour tout élément e_i qui n'est pas le dernier élément de L, son successeur e_j vérifie $e_i < e_j$.

La liste L est accessible via un pointeur p pointant sur son premier élément. Soit n le nombre d'éléments de L.

Question 1 *Quelle est la complexité d'une fonction qui supprime l'élément minimum de L. Justifier.*

Question 2 *Quelle est la complexité d'une fonction qui supprime l'élément maximum de L. Justifier.*

Supposons maintenant que L soit une liste doublement chaînée circulaire : chaque élément contient un pointeur vers son successeur et un pointeur vers son prédécesseur; de plus, le dernier élément de L a pour successeur le premier élément et le premier élément a pour prédécesseur le dernier élément.

La liste L est ordonnée et est accessible via un pointeur p pointant sur son premier élément. Soit n le nombre d'éléments de L.

Question 3 *Quelle est la complexité d'une fonction qui supprime l'élément minimum de L. Justifier.*

Question 4 *Quelle est la complexité d'une fonction qui supprime l'élément maximum de L. Justifier.*

Question 5 *Comparer les deux implémentations proposées pour L.*