Université Paris 7 - Licence d'informatique - L2

Eléments d'Algorithmique

Examen du 26 Mai 2 1 durée : 2h

La qualité de la rédaction sera prise en compte dans la notation. Justifiez toutes vos réponses et expliquez les fondements de vos algorithmes en français avant de les rédiger en pseudo-code compréhensible et commenté où nécessaire. Le barème est donné seulement à titre indicatif.

Exercice 1 - Tas (3 points).

Donnez une implémentation détaillée de l'opération `delete(T : tas, i : entier)` dans un tas-min binaire implémenté dans le type `tas` présenté en cours, qui retourne un tas-min dans lequel l'élément situé à l'indice i dans le vecteur a été supprimé. Attention aux effets de bord au niveau des feuilles et au problème des noeuds avec un seul fils !

Exercice 2 - AVL (4 points)

On rappelle que dans un AVL chaque nœud contient un champs *bal* (pour balance) dont la valeur marque la différence de hauteur entre les sous-arbres du noeud.

1. Dessinez la suite des arbres AVL construits par insertions successives des entiers $5, 6, 8, 2, 1, 9, 7, 3, 4$ dans un AVL initialement vide, en expliquant les étapes des insertions. On souhaite voir précisément le résultat de l'insertion dans l'arbre avant les rotations, les valeurs des balances, la suite des rotations qui mène à l'AVL.
2. Dans cet AVL donner le résultat et les étapes intermédiaires de la suppression des valeurs : 5 et 9.

Exercice 3 - Tri Rapide (5 points)

Simuler l'algorithme de tri rapide vu en cours pour trier le tableau : $4, 7, 2, 3, 6, 8, 1, 9, 5$. On détaillera chaque application de la procédure `Partition` en sachant qu'elle choisit comme pivot l'élément de trouvant au début de la partie du tableau à partitionner. On pourra représenter avec un graphe l'imbrication des appels récursifs effectués, ainsi que les tableaux qu'ils retournent.

Problème 4 - Algorithmes et complexité (8 points)

Soit T un tableau de n boules, dont certaines sont bleues et d'autres rouges (le nombre de boules rouges et de boules bleues n'est pas connu et n'est pas déterminant pour la suite du problème).

a) Ecrire une fonction itérative (en pseudo-code) qui répartit le tableau de manière à ce que toutes les boules bleues soient placées devant les boules rouges. La fonction proposée doit travailler sur place (c'est à dire, sans utiliser de tableau auxiliaire), doit effectuer uniquement des déplacements de boules et doit avoir une complexité en temps en $O(n)$ et en espace $O(1)$. On n'est pas autorisé à effectuer un comptage des boules bleues ou des boules rouges.
On rappelle que pour la compléxité en éspace, seulement les variables de travail sont prises en compte, alors que les données (ici, le tableau) ne le sont pas.

Soit maintenant T un tableau de n boules, dont certaines sont bleues, d'autres blanches et d'autres rouges (le nombre de boules de chaque couleur n'est pas connu et n'est pas déterminant pour la suite du problème).

b) Ecrire une fonction itérative (en pseudo-code) qui répartit le tableau de manière à ce que toutes les boules bleues soient placées devant, suivies des boules blanches, et enfin les boules rouges, pour former le drapeau français. Comme la précédente, cette fonction doit travailler sur place, doit effectuer uniquement des déplacements de boules et doit avoir une complexité en temps en $O(n)$ et en espace $O(1)$. On n'est pas autorisé à effectuer un comptage des boules de chaque couleur, ni à changer la couleur d'une boule.

Suggestion. Utiliser trois indices `blu`, `bla`, `rou`. S'assurer que votre algorithme préserve l'invariant suivant :
- `blu` $\leq$ `bla` $\leq$ `rou`.
- Pour tout i avec $0 \leq i < blu$, on a T[i] = bleu.
- Pour tout i avec $blu \leq i < bla$, on a T[i] = blanc.
- Pour tout i avec $rou \leq i < n$, on a T[i] = rouge.
- Pour tout i avec $bla \leq i < rou$, on a T[i] = indéterminé.

Ceci est illustré par la figure suivante :

On peut, bien sûr, envisager d'autres solutions, mais il est impératif qu'elles respectent les conditions de la question b).

c) Si l'invariant précédent s'applique à votre algorithme, faites-en la preuve, sinon proposez-en un qui démontre la validité de votre algorithme.

d) Si l'incrémentation ou la decrémentation d'un index coute p et l'échange de deux boules coute $3q$, quelle est la fonction de complexité dans le meilleur cas en termes de ces opérations ? Et dans le pire des cas ?

e) Proposez un autre algorithme en $O(n)$ et en espace $O(1)$, si on s'autorise aussi le comptage des boules et le changement de couleur d'une boule.

f) Si l'incrémentation d'un compteur coute p et le changement de couleur d'une boule coute q quelle est la fonction de complexité de l'algorithme proposé à la question e) dans le meilleur cas en termes de ces opérations ? Et dans le pire des cas ?

g) Quelle est la condition sur p et q pour déterminer lequel des deux algorithmes on a intérêt à choisir dans le meilleur cas ? Et dans le pire des cas ?