
Devoir à rendre le 25 mai 2009

Le barème est uniquement indicatif.

Vous pouvez écrire les algorithmes en C ou en pseudo-code. Si cela vous pose trop de difficultés n'hésitez pas à répondre en décrivant un algorithme par des phrases ! Vous pouvez faire appel à des fonctions auxiliaires vues en cours (comparaison, tris, sous-tableau, etc.).


Première partie

10 points

Exercice 1 (Partiel mai 2008).

Rappeler les définitions utilisées et justifier (démontrer) vos réponses.

1. Est-ce que $n^2 - 2n + 1 = O(n^2)$?
2. Est-ce que $\sum_{i=1}^n \log i = \Omega(n)$?
3. Est-il vrai que si $f = O(g)$ et $f = \Omega(h)$ alors $g = \Omega(h)$?


 1 pt
9 min

 1 pt
9 min

 1 pt
9 min

Exercice 2 (Invariant de boucle).

Étant donné un tableau T de $N > 0$ entiers, l'algorithme Maximum renvoie l'indice de l'élément maximum de T . Démontrer le à l'aide d'un invariant de boucle.


 1.5 pt
13 min

Fonction Maximum(T)

```
m = 0;
pour i ← 1 à Taille(T) - 1 faire
    si T[i] ≥ T[m] alors
        m = i;
retourner m;
```


Exercice 3 (Piles, files).


Partant d'une pile vide, on ajoute (empile) 10, puis 20, 30, 40, 50 combien doit-on retirer (dépiler) d'éléments pour que le prochain élément qui sera retiré soit 40 ? (Facile) Même question en utilisant une file (initialement vide).

 0,5 pt
4 min

Exercice 4 (tas).


1. Former un tas max en insérant successivement les éléments 5, 7, 3, 9, 2, 0, 6, 4, 8, 1, répondre en représentant le tas obtenu.
2. Dans le photocopié du cours vous trouverez une autre façon de former un tas. Essayez là sur l'exemple précédant (vous considérerez que les éléments de la question précédente sont rangés dans le même ordre dans un tableau). Cette question ne compte pas pour le devoir !
3. Supprimer l'élément maximum du tas, répondre en représentant le nouveau tas.

 1 pt
9 min

 0.5 pt
4 min


Exercice 5 (Insertion / suppression ABR).

Former un arbre binaire de recherche en insérant successivement et dans cet ordre les éléments : 5, 7, 3, 9, 2, 0, 6, 4, 8, 1 (répondre en représentant l'arbre obtenu). Supprimer l'élément 5. (Répondre en représentant le nouvel arbre. Il y a deux réponses correctes possibles, selon la variante choisie pour l'algorithme de suppression).

 1 pt
9 min


Exercice 6.

Écrire une fonction récursive Somme(x) prenant un arbre binaire dont les éléments sont des entiers et rendant la somme de ses éléments.

 1 pt
9 min

Exercice 7.

Classer les fonctions de complexité $n \log n$, $\log n$, n^2 , n^2 , n par ordre croissant. Pour les complexités $\log n$ et n donner l'exemple d'un algorithme (du cours ou des TD) qui a asymptotiquement cette complexité en pire cas, en temps.

 1.5 pt
13 min

Problème du rang (cinq exercices)

tot: 10 pt

Dans cette partie on s'intéresse au problème suivant : étant donné un ensemble A de N éléments deux à deux comparables, déterminer quel est l'élément de rang k (le k -ième plus petit élément), c'est à dire l'élément $x \in A$ tel que exactement $k - 1$ éléments de A sont plus petits que x . Bien entendu, on peut supposer que k est choisi tel que $1 \leq k \leq N$. On pourra considérer que les éléments de A sont distincts (la comparaison ne les déclare jamais égaux). Si on a par exemple les éléments 23, 62, 67, 56, 34, 90, 17 (N vaut 7) alors l'élément de rang 3 est 34.

Exercice 8.

Dans la liste 21, 90, 30, 18, 35, 45, 77, quel est l'élément de rang 4 ?

0,5 pt
4 min

Les exercices suivants portent sur l'écriture d'un algorithme réalisant la fonction de recherche de l'élément de rang k dans un ensemble A de N éléments : Sélection(S, k), où S est la structure de donnée contenant les éléments de A .

Rang dans un tableau

On suppose que les éléments de A sont donnés en entrée dans un tableau T non trié de N éléments.

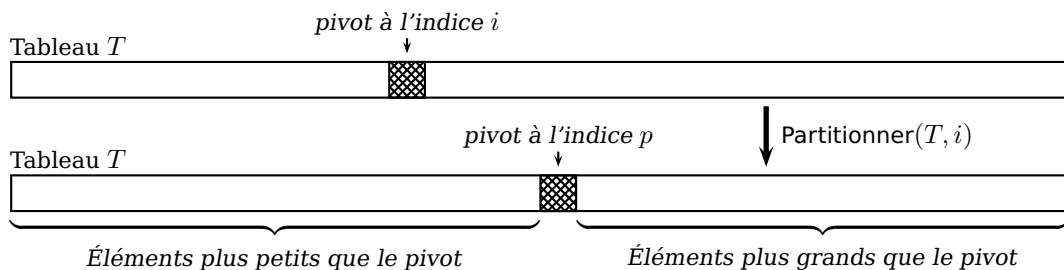
Exercice 9.

Quel est le moyen *le plus simple* de réaliser la fonction Sélection(T, k) ? La réponse doit tenir en 3-4 lignes de pseudo-code ou de C en faisant appel (comme fonction auxiliaire) à un algorithme du cours dont vous préciserez le nom. Donner un minorant asymptotique en fonction de N en pire cas du temps d'exécution de Sélection(T, k).

1 pt
9 min

Rappel sur le tri rapide. On rappelle le fonctionnement du tri rapide (*quicksort*), algorithme permettant de trier un tableau d'éléments deux à deux comparables (le résultat, c'est à dire le tableau contenant les éléments dans l'ordre est rendu en place). Le principe de fonctionnement est le suivant. Si le tableau contient un ou zéro élément, il n'y a rien à faire. Si le tableau contient plus d'un élément :

Partition on choisit un élément du tableau, d'indice i , appelé *pivot* et on partitionne le tableau, en place, autour de cet élément. Après partition, le pivot est à un certain indice p du tableau, les éléments plus petits que le pivot sont (éventuellement dans le désordre) aux indices inférieurs à p et les éléments plus grands que le pivot sont (éventuellement dans le désordre) aux indices supérieurs à p . Le pivot est donc à sa place définitive.







Appels récursifs On relance le tri sur chacun des deux sous-tableaux obtenus par partition : le tableau des éléments d'indices inférieurs à p et le tableau des éléments d'indices supérieurs à p .

Exercice 10.

Dans cet exercice, on va mettre en œuvre un algorithme inspiré du tri rapide pour réaliser la fonction Sélection(T, k). L'idée est qu'il n'est pas besoin de trier tout le tableau pour trouver l'élément de rang k .

On suppose que les indices du tableau commencent à 0. La remarque importante est que le pivot après partitionnement est l'élément de rang $p + 1$. Si après partition on trouve un p égal à $k - 1$, alors l'élément de rang k est le pivot.

On suppose que l'on a une fonction $\text{Partitionner}(T, i)$ qui effectue la partition autour d'un élément d'indice i (pivot) et renvoie l'indice p du pivot après partition.

1. Après partition autour d'un pivot où chercher l'élément de rang k lorsque $k < p + 1$? Et lorsque $k > p + 1$?  0.5 pt
4 min
2. Décrire simplement le principe d'un algorithme récursif, basé sur la partition, réalisant $\text{Sélection}(T, k)$. (A t-on besoin de faire deux appels récursifs comme dans le tri rapide?)  1 pt
9 min
3. Écrire en pseudo-code ou en C, l'algorithme $\text{Sélection}(T, k)$ sous forme itérative.  2 pt
18 min
4. Écrire l'algorithme de partitionnement $\text{Partitionner}(T, i)$. On pourra commencer par échanger l'élément d'indice i avec l'élément d'indice 0 de manière à se ramener toujours au cas où le premier élément du tableau a été choisi comme pivot.  1.5 pt
13 min

Rang dans un arbre binaire de recherche avec comptage des descendants

On enrichit la structure de données *arbre binaire de recherche* (ABR) en ajoutant à chaque nœud x un entier $\text{Total}(x)$ donnant le nombre d'éléments stockés dans le sous-arbre dont le nœud x est racine, l'élément stocké dans x inclus. Exemple figure 1.

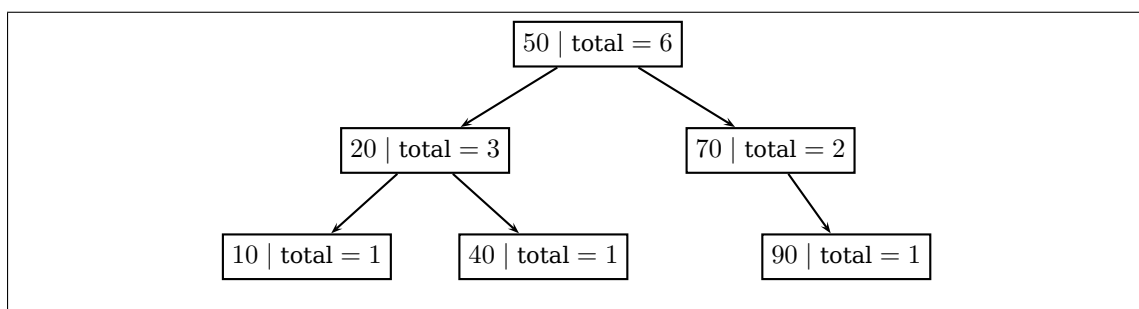



Fig. 1: Un exemple d'arbre binaire de recherche avec comptage des descendants

Dans la suite, on suppose que les éléments de A sont stockés dans un ABR (de racine r) enrichie par ce comptage.

Exercice 11.

Écrire un algorithme (C ou pseudo-code) réalisant la fonction $\text{Sélection}(r, k)$. Donner en fonction de la hauteur h de l'arbre, un majorant asymptotique serré de son temps d'exécution.  1.5 pt
13 min

En guise d'aide, on rappelle, en pseudo-code, l'algorithme de recherche dans un ABR.

Fonction Rechercher(r, c)

Entrées : Le nœud racine r d'un arbre binaire de recherche et une clé c .

Sorties : Le nœud de l'arbre contenant l'élément de clé c s'il en existe un, ou le nœud vide N sinon.

si EstVide(r) **alors**
| retourner N ;

sinon

si $c = \text{Clé}(r)$ **alors**
 | retourner r ;

sinon



si $c < \text{Clé}(r)$ **alors**
 | retourner Rechercher(Gauche(r), c);

sinon

 | retourner Rechercher(Droite(r), c);

Exercice 12.

Pour réaliser les ABRs avec comptage des descendants, il faut adapter les fonctions d'ajout et de suppression d'un élément.

1. Rappeler (en C ou en pseudo-code) l'algorithme d'ajout d'un élément dans un ABR et indiquer les modifications à lui apporter.  1 pt
9 min
2. Pour la suppression d'un élément, il ne vous est pas demandé de rappeler le code de la fonction la réalisant. Mais supposons que cette fonction renvoie une référence sur le parent p du nœud qui a été *réellement* supprimé. Donnez dans ce cas une méthode (une fonction prenant le paramètre p) pour mettre à jour le comptage à travers l'arbre.  1 pt
9 min