Examen 2I003

Jeudi 17 décembre 2015, 2 heures
aucun document autorisé

Exercice 1 – Arbres binaires et tas - 4.5 points

Dans cet exercice, T désigne un arbre binaire, $n(T)$ son nombre de nœuds et $h(T)$ sa hauteur.

Question 1

Donnez la définition inductive de l'ensemble AB des arbres binaires dont les nœuds sont étiquetés par des éléments de $\mathcal{A}$. On rappelle que l'arbre vide est un arbre binaire. Donnez la définition (non inductive) d'un arbre parfait et d'un tas.

Question 2

Démontrez par induction sur la définition inductive de AB que, pour tout $T \in AB$, $h(T) \leq n(T) \leq 2^{h(T)} - 1$. En déduire que, pour tout arbre parfait T, $2^{h(T)-1} \leq n(T) < 2^{h(T)}$.

Question 3

1. Triez par ordre décroissant en utilisant un tri par tas les valeurs de la liste $L = (37, 42, 10, 2, 28, 16)$. Vous donnerez uniquement les valeurs successives du tableau courant après chaque opération de construction ou de destruction du tas.
2. Quelle est la complexité du tri par tas dans le pire des cas ? Justifiez votre réponse.

Exercice 2 – Graphes et arbres - 4 points

Dans cet exercice, $G = (V, E)$ est un graphe non orienté.

Question 1

Rappelez la définition du degré $d_G(x)$ pour tout $x \in V$. Calculer $d_G(x)$ pour le graphe $G = (V, E)$ avec $V = \{1, 2, 3, 4, 5\}$ et $E = \{\{1, 2\}, \{2, 3\}, \{1, 3\}, \{2, 4\}, \{3, 4\}, \{4, 5\}\}$.

Question 2

Démontrez par récurrence sur le nombre d'arêtes que, pour tout graphe non orienté $G = (V, E)$, $\sum_{x \in V} d_G(x) = 2|E|$. En déduire que tout graphe possède un nombre pair de sommets de degré impair.

Question 3

Quelle est la définition d'un graphe connexe ? Quelle est la définition d'un graphe minimal connexe ? Quelle est la définition d'un arbre ?

Question 4

Démontrez par l'absurde que, si G est un arbre, alors G est minimal connexe. Que pensez-vous de la réciproque ? Justifiez votre réponse.

Exercice 3 – Tri par sélection du min et du max - 12 points

On rappelle que l'indexation des tableaux commence à 0. Si $T = [a_0, \ldots, a_i, \ldots, a_j, \ldots a_{n-1}]$, on note $T[i..j] = [a_i, \ldots, a_j]$. Dans tout l'exercice, T est un tableau d'entiers dont la taille est notée n.
On dispose de la fonction `indMinMax(T,g,d)` ainsi définie, pour $0 \leq g \leq d \leq n-1$:

```
def indMinMax(T, g, d):
    a = g ; z = g ; i = g + 1
    while i <= d:
        if T[i] < T[a]:
            a = i
        elif T[i] >= T[z]:
            z = i
        i = i + 1
    return (a, z)
```

La fonction `indMinMax(T,g,d)` renvoie le couple (a, z) où a est le plus petit indice appartenant à $[g..d]$ tel que $T[a]$ est le minimum de $T[g..d]$ et z est le plus grand indice appartenant à $[g..d]$ tel que $T[z]$ est le maximum de $T[g..d]$.

Question 1

1. On considère le tableau `T0 = [6,8,1,7,4,3,9,5]`. Exécuter l'appel de`indMinMax(T0,0,7)`, en donnant les valeurs de a, z et i à la fin de chaque itération, ainsi que la valeur retournée par la fonction.
2. Calculer le nombre minimum de comparaisons entre éléments du tableau effectuées par `indMinMax(T,g,d)`. Donner un exemple de tableau pour lequel `indMinMax` effectue le minimum de comparaisons.
3. Calculer le nombre maximum de comparaisons entre éléments du tableau effectuées par `indMinMax(T,g,d)`. Donner un exemple de tableau pour lequel `indMinMax` effectue le maximum de comparaisons.
4. Calculer le nombre maximum d'affectations (aux variables a et z) effectuées par `indMinMax(T,g,d)`. Donner un exemple de tableau pour lequel `indMinMax` effectue le maximum d'affectations.

On dispose aussi d'une procédure `permuter(T,g,d,a,z)`, pour $0 \leq g < d \leq n-1$, $g \leq a \leq d$, $g \leq z \leq d$ et $a \neq z$, qui transforme le tableau T en un tableau T' tel que :

— $T'[g..d]$ est une permutation de $T[g..d]$
— $T'[g] = T[a]$ et $T'[d] = T[z]$
— $T'[0..(g-1)] = T[0..(g-1)]$ et $T'[(d+1)..(n-1)] = T[(d+1)..(n-1)]$.

On considère la procédure `placerMinMax(T,g,d)` ainsi définie, pour $0 \leq g < d \leq n-1$:

```
def placerMinMax(T,g,d):
    (a, z) = indMinMax(T,g,d)
    print('a=', a, 'z=', z)
    permuter(T, g, d, a, z)
    print('T[',g,']=', T[g], 'T[',d,']=', T[d])
```

Question 2

1. On considère le tableau `T0=[6,8,1,7,4,3,9,5]` et l'appel `placerMinMax(T0,1,5)`. Donner les affichages de cet appel.
   Remarque : il est inutile de connaître en détail l'ordre des éléments de $T[g+1..d-1]$ après l'appel `permuter(T,g,d,a,z)` pour répondre à cette question.
2. Notons T' le tableau obtenu après l'exécution de `placerMinMax(T,g,d)`, pour $0 \leq g < d \leq n-1$. Montrer que :
   — $T'[g..d]$ est une permutation de $T[g..d]$,
   — $T'[g]$ est le plus petit élément de $T'[g..d]$,
   — $T'[d]$ est le plus grand élément de $T'[g..d]$,
   — $T'[0..(g-1)] = T[0..(g-1)]$ et $T'[(d+1)..(n-1)] = T[(d+1)..(n-1)]$.
3. On admet que `permuter(T,g,d,a,z)` n'effectue aucune comparaison entre éléments du tableau et effectue un nombre d'affectations aux éléments de T en $\Theta(1)$. Quel est le nombre de comparaisons entre éléments du tableau effectuées par `placerMinMax(T,g,d)` ? Quel est le nombre d'affections aux éléments du tableau effectuées par `placerMinMax(T,g,d)` ?

On considère la procédure `trierGD(T, g, d)` ainsi définie, pour $0 \leq g \leq d \leq n-1$:

```
def trierGD(T, g, d):
    if g < d:
        print('appel␣avec␣g=', g, 'et␣d=',d)
        placerMinMax(T, g, d)
        trierGD(T, g + 1, d - 1)
        print('retour␣T␣=', T)
```

Question 3

Exécuter l'appel de `trierGD(T0,0,7)`, avec `T0=[6,8,1,7,4,3,9,5]`, en donnant les affichages successifs (ne pas oublier les affichages faits par `placerMinMax(T,g,d)` !).

Question 4

Montrer, par récurrence sur $d-g$, que la procédure `trierGD(T,g,d)` trie le sous-tableau $T[g..d]$ en ordre croissant et laisse les autres éléments de T inchangés.

Question 5

En utilisant la procédure `trierGD(T,g,d)`, définir une procédure `trier(T)` qui trie le tableau `T` en ordre croissant. Justifier la réponse.

Question 6

1. Montrer que le nombre d'affectations aux éléments du tableau effectuées par `trier(T)` est en $\Theta(n)$.
2. Montrer que le nombre de comparaisons entre éléments du tableau effectuées par `trier(T)` est en $\Theta(n^2)$.