Algorithmique avancée : Partiel du Jeudi 26 Octobre 2017.

Durée 2h. Seul document autorisé : 1 feuille A4 recto-verso.

Le barème est donné à titre indicatif. Toutes les réponses doivent être soigneusement **justifiées**. La compréhension du sujet faisant partie de l'épreuve, **on ne répondra à aucune question**. Si vous rencontrez des ambiguïtés, vous expliquerez **sur votre copie** comment vous les interprétez.

I - Médiane de deux tableaux (17 points)

On appelle médiane d'un ensemble d'entiers, un élément qui possède autant d'éléments supérieurs ou égaux que d'éléments inférieurs ou égaux dans cet ensemble.

Soient $A[1..m]$ et $B[1..m]$ deux tableaux triés par ordre croissant. On cherche à trouver une médiane de l'ensemble des éléments de ces deux tableaux.

L'algorithme suivant retourne une médiane.

Function $\textsc{Mediane}(A[1..m], B[1..m])$

Require: A, B : `sorted arrays of size` m
- **if** $m = 1$ **then**
  - **if** $A[1] \leq B[1]$ **then return** $A[1]$ **else return** $B[1]$
- **else**
  - $x \leftarrow [(m+1)/2]$
  - **if** $A[x] \leq B[x]$ **then**
    - **return** $\textsc{Mediane}(A[(x+1)..m], B[1..[m/2]])$
  - **else**
    - **return** $\textsc{Mediane}(A[1..[m/2]]), B[(x+1)..m]$

On considère les deux tableaux suivants :

$A =$

7	28	46	53	64	84	89

et $B =$

3	12	14	21	45	78	122

A) Premier algorithme (3 points)

1. Faîtes tourner l'algorithme avec l'appel $\textsc{Mediane}(A[1..7], B[1..7])$ et vérifiez que l'élément retourné est bien la médiane.

   On note $S_{\textsc{Mediane}}(n)$ et $T_{\textsc{Mediane}}(n)$ les complexités respectivement spatiales et temporelles en pire cas de l'algorithme $\textsc{Mediane}$ appliqué à des données de taille totale n.

2. En supposant que les appels à $\textsc{Mediane}$ sur les sous-tableaux ne recopient pas les tableaux mais utilisent simplement des décalages des indices de début et de fin de ces tableaux, donnez l'expression de la complexité spatiale en pire cas de ce programme en notation asymptotique c'est-à-dire proposez une fonction g telle que $S_{\textsc{Mediane}} \in \Theta(g)$.

3. Donnez l'expression de la complexité temporelle en pire cas de ce programme en notation asymptotique Θ c'est-à-dire proposez une fonction g' telle que $T_{\textsc{Mediane}} \in \Theta(g')$.

B) Médiane avec un tas binaire (7 points)

Sachant que le m-ième élément par ordre croissant d'un ensemble de $2m$ éléments en est une médiane, on propose l'algorithme suivant :

```
Function MedianeTas(A[1..m], B[1..m])
  Require: A, B : arrays of size m
    for i = 1 to m do
        C[i] = A[i]
        C[i + m] = B[i]
    BuildHeap(C)
    for i = 1 to m do r ← Remove(C)
    return r
```

où BuildHeap(T) est la fonction qui construit un tas binaire minimal en-place à partir d'un tableau T et Remove(T) est une fonction qui supprime et renvoie la racine du tas T.

1. On lance l'algorithme avec l'appel MedianeTas(A[1..7],B[1..7]). Décrivez l'évolution du tableau C après le BuildHeap et après les deux premiers Remove.
2. En vous servant des complexités des fonctions de traitement des tas binaires vues en cours, exprimez les complexités spatiale $S_{\text{MedianeTas}}(n)$ et temporelle $T_{\text{MedianeTas}}(n)$ en pire des cas de cet algorithme en fonction de la taille n des données, vous donnerez donc deux fonctions h et h' telles que $S_{\text{MedianeTas}} \in \Theta(h)$ et $T_{\text{MedianeTas}} \in \Theta(h')$.

C) Médiane avec un tas binomial (7 points)

On décide maintenant d'utiliser un tas binomial pour profiter de la fonction qui fusionne deux tas binomiaux. Voici l'algorithme utilisé.

```
Function MedianeTasBinom(A[1..m], B[1..m])
  Require: A, B : arrays of size m
    H_A ← CreateBinomialHeap()
    H_B ← CreateBinomialHeap()
    for i = 1 to m do
        Add(H_A,A[i])
        Add(H_B,B[i])
    H ← Merge(H_A,H_B)
    for i = 1 to m do r ← RemoveMin(H)
    return r
```

où CreateBinomialHeap() est la fonction qui crée un tas binomial vide, Add(H,e) ajoute l'élément e au tas H, Merge(H_1, H_2) retourne un tas résultant de la fusion de deux tas, RemoveMin(H) supprime et retourne l'élément minimum du tas H.

1. Faîtes tourner l'algorithme avec l'appel MedianeTasBinom(A[1..7],B[1..7]) afin de décrire les tas binomiaux H_A et H_B obtenus après la boucle **for** et le tas binomial H issu de Merge, puis le tas H résultant du premier RemoveMin.
2. En vous servant du fait que la complexité de CreateBinomialHeap est en $\Theta(1)$ et des complexités des fonctions de traitement des tas binomiaux vues en cours, exprimez la complexité temporelle $T_{\text{MedianeTasBinom}}(n)$ en pire des cas de cet algorithme en fonction de la taille n des données, vous donnerez donc une fonction h telle que $T_{\text{MedianeTasBinom}} \in \Theta(h)$.
3. (Bonus) Comparez les complexités des trois méthodes en tenant compte de la complexité du tri des deux tableaux initiaux nécessaire pour effectuer la première méthode.

II - B-arbre des 100 premiers entiers (3 points)

On crée un B-arbre de degré 4 en insérant les 100 premiers entiers dans l'ordre croissant.

1. Quelle est la hauteur de cet arbre ?
2. Quelle est la première clé de sa racine ?