Structures de données et algorithmes

Répétition 4: Tas, file à priorité et arbre

Romain Mormont

27 mars 2020

Exercice 1

(a) Combien y a-t-il au minimum (resp. maximum) d'éléments dans un tas de hauteur h ?

(b) Où l'élément le plus petit réside dans un tas-max ?

(c) Quelle est la relation entre un tas-min et un tableau trié (par ordre croissant) ?

(d) Est-ce que le tableau $[23, 17, 14, 6, 13, 10, 1, 5, 7, 12]$ est un tas-max ?

Exercice 2

Dessiner tous les tas-min possibles avec l'ensemble des clés $\{A, B, C, D, E\}$, où chaque clé n'apparaît qu'une seule fois.

Exercice 3

Soit un tas-min `H`. Illustrer l'exécution des opérations suivantes :

```
heapInsert(H, 5)
heapInsert(H, 7)
heapInsert(H, 4)
heapInsert(H, 8)
heapExtractMin(H)
heapInsert(H, 3)
heapInsert(H, 4)
heapInsert(H, 10)
heapExtractMin(H)
heapInsert(H, 3)
```

Exercice 4

(a) Comment implémenter une pile au moyen d'une file à priorité ?

(b) Comment implémenter une file au moyen d'une file à priorité ?

(c) Comment implémenter une file aléatoire au moyen d'une file à priorité ?

(d) Comment implémenter une file à priorité maximum avec un tas-min ?

(e) Comment implémenter une file à priorité qui soit $\Theta(1)$ pour l'insertion et $O(n)$ pour l'extraction ?

Exercice 5

Le parcours préfixe d'un arbre binaire donne :

```
A B C - - D - - E - F - -
```

En donner les parcours infixe et postfixe. Les lettres correspondent aux noeuds internes et les tirets à des feuilles de l'arbre.

Exercice 6

Soit un arbre `T`.

(a) Ecrire une méthode `size(T)` qui détermine le nombre d'éléments dans l'arbre `T`.

(b) Ecrire une méthode `isBalanced(T)` qui détermine si l'arbre `T` est équilibré
Note : Un arbre est équilibré si pour chacun de ses nœuds, la hauteur du sous-arbre de droite de ce nœud ne diffère pas de la hauteur du sous-arbre de gauche de plus d'une unité ;

(c) Ecrire une méthode `isComplete(T)` qui détermine si l'arbre `T` est complet ou non.

(d) Ecrire une méthode `mirror(T)` qui retourne l'arbre miroir de l'arbre `T`.

(e) Ecrire une méthode `printLowerThan(T, x)` qui imprime les valeurs contenues dans `T` inférieures ou égales à x. Quelle est la complexité de cette opération ?

Bonus : Pour chaque fonction, décrivez un algorithme a) récursif et b) non-récursif.

Bonus : Pourquoi les implémentations récursives sont faciles ?