L3-INFO Algorithmique

Quick 2 – 10 novembre 2023 – durée 1h

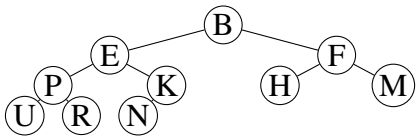
Les documents et les téléphones (incluant smartphone, tablettes,... tout ce qui contient une interface réseau) sont interdits. Les calculettes sont autorisées.
Seuls les dictionnaires pour les personnes de langue étrangère sont autorisés.
Toutes les réponses doivent être justifiées. Le barème est indicatif.

---

1 Premiers nœuds dans un tas (*6 points*)

Dans cet exercice, on considère des tas-min (où chaque nœud contient une clé **inférieure** à celles de ses fils), représentés sous forme de tableaux indexés à partir de 0, comme vu en TD.
Les différentes opérations sur les tas **ne sont pas supposées implémentées**, il faudra donc les réécrire vous-mêmes si vous pensez qu'elles sont utiles.
Pour simplifier, on pourra supposer que tous les éléments du tas sont distincts.

Pour tous les exemples donnés dans l'énoncé, on utilisera le tas :



Question 1 : *(1 pt)*

On cherche à déterminer le 2è plus petit élément dans un tas (contenant au moins 2 éléments).
Dans notre exemple, il faudrait renvoyer E.
Donnez un algorithme qui résout ce problème en temps constant.

Question 2 : *(2 pts)*

On cherche à déterminer le 3è plus petit élément dans un tas. On supposera pour cette question que le tas comporte largement plus de 3 éléments (au moins 10 par exemple).
Dans notre exemple, il faudrait renvoyer F.
Donnez un algorithme qui résout ce problème en temps constant.

Question 3 : *(3 pts)*

Donnez un algorithme qui renvoie le plus **grand** élément dans un tas.
Dans notre exemple, il faudrait renvoyer U.
Donnez un algorithme qui résout ce problème avec une complexité $\mathcal{O}(n)$.

2 Preuve d'algorithme *(14 points)*

Soit un tableau T (indexé de 0 à $n-1$). On cherche à inverser l'ordre des éléments présents dans T. Par exemple, pour $T = [E, K, S, G, N, P]$, l'objectif est de transformer ce tableau pour que $T = [P, N, G, S, K, E]$.

On propose l'algorithme suivant :

INVERSER(T, n)
Données : Un tableau T de n éléments quelconques
Résultat : Le tableau T est modifié pour inverser l'ordre de ses éléments

```
1 i := 0
2 j := n − 1
3 tant que i < j faire
4 |   x := T[i]
5 |   T[i] := T[j]
6 |   T[j] := x
7 |   i := i + 1
8 |   j := j − 1
```

Question 4 : *(2 pts)*

Justifiez que cet algorithme se termine.

Question 5 : *(2 pts)*

On appelle T le tableau initial, et T' le tableau obtenu en fin d'exécution de l'algorithme.

a. Représentez sous forme de schéma la postcondition qui exprime que notre algorithme réalise la tâche souhaitée.
b. Exprimez également cette postcondition en faisant référence explicitement à des indices des tableaux T et T' (soit par une phrase en français, soit par une formule de la forme $\forall i \ldots T'[i] \ldots$).

Question 6 : *(3 pts)*

a. Exprimez un invariant vérifié par la boucle Tant Que de notre algorithme qui permettra de démontrer cette postcondition (encore une fois, en français ou comme une formule, du moment que les indices concernés sont explicites).
b. Présentez également votre invariant sous forme d'un schéma.

Question 7 : *(1 pt)*

Justifiez que cet invariant est vérifié après la ligne 2 de l'algorithme.

Question 8 : *(3 pts)*

Démontrez que la boucle Tant Que maintient effectivement cet invariant.

Question 9 : *(2 pts)*

Démontrez enfin la postcondition à l'aide de l'invariant en sortie de boucle.

Question 10 : *(1 pt)*

Si la condition de la boucle Tant Que était $i < n$, notre algorithme serait incorrect. Expliquez quelle étape de la démonstration que nous venons d'écrire ne serait plus valable.