Université de Mons
FS/1/5684 – Algorithmique
Examen de première session – Partie théorique

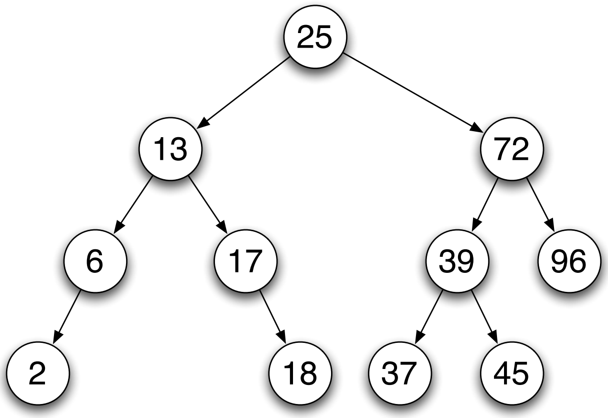
Le 22 janvier 2010

Consignes

- Pour cette partie, vous n'avez pas le droit d'utiliser de notes.
- Cette partie de l'examen dure 1 heure 15 minutes.
- Veillez à bien justifier vos réponses. Une réponse mal justifiée, même correcte, ne permet pas d'obtenir le maximum des points.
- Quand vous indiquez une complexité, veillez à bien expliquer ce que sont les paramètres qui apparaissent dans le $\mathcal{O}$. Par exemple, $\mathcal{O}(n^2)$ n'a aucun sens si n n'apparaît pas dans l'algorithme ou dans la définition de la structure qui est traitée…

Question 1 – 6 points

1. Donnez l'algorithme de *parcours en largeur* (aussi appelé *parcours par niveaux*) d'un arbre binaire. Décrivez brièvement les éventuelles structures de données utilisées.
2. Quelle est la complexité de cet algorithme ? Justifiez en tenant compte des complexités des opérations sur les éventuelles structures de données utilisées par l'algorithme.
3. La complexité de cet algorithme est-elle améliorée si on fait l'hypothèse que l'arbre est équilibré ? Justifiez.
4. Appliquez cet algorithme sur l'arbre ci-dessous et donnez l'ordre dans lequel il en parcourt les nœuds.



Correction

Voir syllabus Section 7.2.4 (pp. 116 *sqq.*). Cet algorithme utilise une file comme structure, qui a la propriété FIFO et dont toutes les opérations peuvent être implémentées en $\mathcal{O}(1)$; voir syllabus Section 6.2 (pp. 83 *sqq.*). Sa complexité est en $\mathcal{O}(n)$, où n est le nombre de nœuds de l'arbre, car toutes les opérations sur la file sont en $\mathcal{O}(1)$. Comme tous les nœuds doivent être parcourus, la structure de l'arbre n'a pas d'effet particulier sur la complexité, et donc, l'algorithme reste en $\mathcal{O}(n)$ si l'arbre est équilibré. Les nœuds de l'arbre donné sont parcourus dans l'ordre : 25, 13, 72, 6, 17, 39, 96, 2, 18, 37, 45.

Question 2 – 4 points

L'algorithme ci-dessous présente la fonction **Facto**. Quelle est sa complexité ? Démontrez.

```
Entier Facto(Entier n)
début
    si n = 1 alors
        retourner 1 ;
    sinon
        n := n × Facto(n − 1) ;
        retourner n ;
fin
```

Correction

Voir syllabus 2008–2009, pp. 45 *sqq*. En particulier, il faut bien faire attention à ce que la preuve par induction établit qu'on fait n appels récursifs quand on appelle la fonction sur la valeur n. La preuve n'établit pas directement que la fonction est en $\mathcal{O}(n)$, mais elle permet de déduire que la fonction est en $\mathcal{O}(n)$.