Université Paris 7 - Licence d'informatique - L2

Eléments d'Algorithmique

Examen du 23 Juin 2010 - durée : 2h30

La qualité de la rédaction sera prise en compte dans la notation. Justifiez toutes vos réponses et expliquez les fondements de vos algorithmes en français avant de les rédiger en pseudo-code compréhensible et commenté où nécessaire. Le barème est donné seulement à titre indicatif.
Les algorithmes données sans commentaires ou sans explications ne seront pas pris en compte par le correcteur.

Exercice 1 - Complexité (5 points)

Considérez les trois bouts de code suivants :

a)

```
i = 1;
while (i<=n) {
    printf("salut");
    i = 2*i;
}
```

b)

```
i  = 1;
while (i<=n) {
  for (j=1; j<=i; j++) {
    printf("salut");
  }
  i = 2*i;
}
```

c)

```
for (j=1; j<=n; j++) {
  i  = 1;
  while (i<=j) {
    printf("salut");
    i = 2*i;
  }
}
```

Pour chacun, calculez (comme fonction de n) le nombre exact de fois que la chaine "salut" est affichée. Vous pourrez tester des petites valeurs de n.

Si f_a, f_b et f_c sont les trois fonctions de complexité des trois programmes, lesquelles des affirmations suivantes sont vraies ? Justifiez.

1. $f_a \in O(f_b)$
2. $f_a \in O(f_c)$
3. $f_b \in O(f_a)$
4. $f_b \in O(f_c)$
5. $f_c \in O(f_a)$
6. $f_c \in O(f_b)$

Exercice 2 - Tas (4 points).

Détaillez les étapes de l'insertion des éléments suivant dans le tas-min initialement vide :

$$866, 400, 730, 232, 641, 123, 356, 507, 395, 88, 467, 167$$

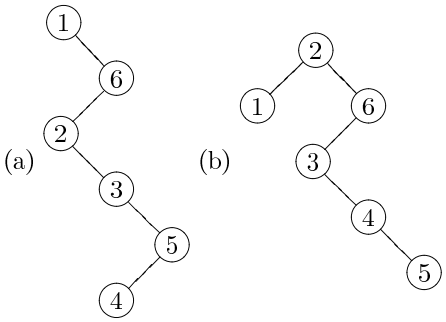
La simulation doit être effectuée sur la représentation des tas qui comprend un tableau d'éléments et un champ entier pour la taille et chaque mouvement des éléments doit être détaillé.

Soit `delete(T : tas, i : entier)` la procédure qui retourne en temps $O(\log n)$ un tas-min dans lequel l'élément situé à l'indice i dans le tas a été supprimé. Simuler l'appel `delete(T, 1)` où T est le tas obtenu à la question précédente. (Le tableau est indexé à partir de 0.)

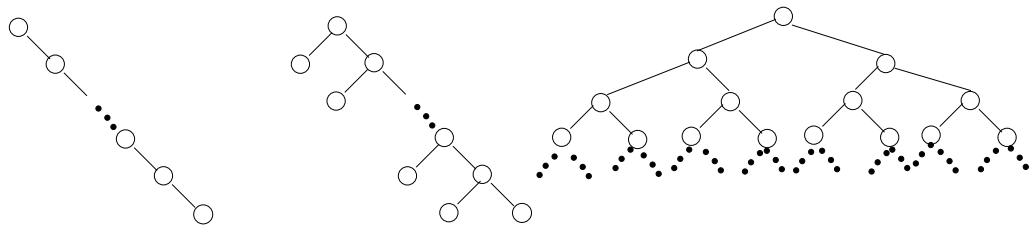
Exercice 3 - Tri (3 points)

Un tableau A de taille n contient des entiers satisfaisant $0 \leq A[i] < \log_2 n$ pour tout i compris entre 0 et $n-1$. Proposez l'algorithme de tri qui vous paraît le plus performant pour trier un tableau de ce type (surtout si on est très pressé d'avoir les résultats...) et justifiez les raisons de votre choix.

Problème 4 - ABR (8 points)



1. Calculer une permutation des entiers $\{1, 2, 3, 4, 5, 6\}$ telle que, si on insère dans un ABR initialement vide les six entiers dans l'ordre de cette permutation, on obtient l'ABR de la figure a). Donner un argument justifiant que cette permutation est unique.
2. Calculer toutes les permutations des entiers $\{1, 2, 3, 4, 5, 6\}$ telles que, si on insère dans un ABR initialement vide les six entiers dans l'ordre de ces permutations, on obtient l'ABR de la figure b).
3. Proposer un algorithme récursif qui reçoit en donnée un ABR et affiche toutes les permutations des éléments contenus dans l'ABR qui auraient pu engendrer l'ABR donné par insértion dans un ABR initialement vide.
4. Calculer le nombre d'appels récursifs effectués par votre algorithme dans les cas suivants :
   - ABR à n noeuds où chaque noeud possède au plus un fils (ABR filiforme)
   - ABR peigne à n noeuds où chaque noeud qui n'est pas une feuille a toujours deux fils, et le fils gauche de chaque noeud (s'il existe) est toujours une feuille.
     (Notez qu'un ABR peigne à n noeuds existe seulement pour certaines valeurs de n. Lesquelles ?)
   - ABR à n noeuds complet parfaitement équilibré.
     (Notez qu'un ABR à n noeuds de ce type existe seulement pour certaines valeurs de n. Lesquelles ?)



ABR filiformes | ABR peigne | ABR complet parfaitement équilibré

   Dans chaque cas, on pourra effectuer des calculs explicites pour des petites valeurs de $n = 1, 2, 3, 4, ...$
5. Déduire de la question 4. une estimation de la complexité en temps dans le meilleur et dans le pire des cas de l'algorithme.
6. Déduire de la question 4. le nombre de permutations engendrant un ABR donné pour chacune des trois familles d'ABR : filiformes, peigne, complets parfaitement équlibrés.