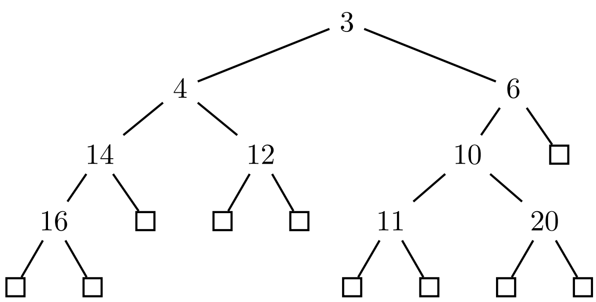
Université Pierre et Marie Curie — **Master Informatique**
Septembre 2006 — **MI015-ALGAV**

Examen
Durée : 2h. Tous documents autorisés.

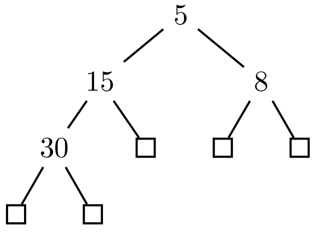
1 Arbres gauches [10 points]

On considère des arbres binaires de taille n (i.e. avec n noeuds internes et $n+1$ noeuds externes -ou feuilles-). Tout noeud interne i a 2 fils, que l'on note $fg(i)$ et $fd(i)$.
On définit le rang d'un noeud de la façon suivante :
- si e est un noeud externe, rang(e) = 0
- si i est un noeud interne, $\text{rang(i)} = 1 + \min(\text{rang(fg(i))}, \text{rang(fd(i))})$.
Un arbre binaire est appelé *gauche*, si pour tout noeud i, $\text{rang(fg(i))} \geq \text{rang(fd(i))}$.
Question 1. Donner les rangs des noeuds de l'arbre gauche suivant.



Question 2. Montrer que dans un arbre gauche, pour tout noeud interne i, si rang(i) = p, le sous-arbre de racine i contient au moins 2^p noeuds externes.
En déduire que pour tout arbre gauche ayant n noeuds internes, la branche droite contient au plus $\lfloor \log_2(n+1) \rfloor$ noeuds internes.

Un tournoi gauche est un arbre gauche dont les noeuds internes portent des clés qui en font un arbre binaire croissant : pour tout noeud i, $\text{cle(i)} \leq \text{cle(fg(i))}$ et $\text{cle(i)} \leq \text{cle(fd(i))}$. (On suppose que les noeuds externes portent tous la clé $+\infty$.) L'arbre ci-dessus est un tournoi gauche.
Les tournois gauches sont une représentation des files de priorité. Pour fusionner deux tournois gauches on utilise la méthode suivante :
Fusionner les branches droites des deux tournois, en arrangeant les noeuds par ordre croissant des clés. Puis recalculer les rangs des noeuds sur le chemin fusionné. Enfin rendre à l'arbre sa forme gauche, en échangeant si besoin les fils droit et gauche des noeuds de ce chemin.
Question 3. Quel est le tournoi gauche résultant de la fusion de l'arbre ci-dessus et de l'arbre suivant :



Question 4. Ecrire l'algorithme de fusion de deux tournois gauches.
Question 5. Quel est l'ordre de grandeur du nombre de comparaisons de clés, dans le pire des cas, pour fusionner deux tournois gauches de tailles respectives n et m.
Question 6. Ecrire l'algorithme de suppression du minimum dans un tournoi gauche, et analyser sa complexité en nombre de comparaisons.
Question 7. Comparer cette représentation des files de priorité avec celles que vous connaissez par ailleurs.

2 Problème de sac à dos [5 points]

On considère le problème du sac-à-dos suivant : un randonneur souhaite organiser un voyage et possède un ensemble de p objets. Chaque objet i possède un poids w_i et une utilité u_i. L'objectif du randonneur est de remplir son sac-à-dos de façon à maximiser sa fonction utilité tout en respectant la contrainte limitant le poids total du sac-à-dos à W.

Question 1. Formuler ce problème sous-forme de problème de plus long chemin dans un graphe orienté que vous définirez pour l'exemple ci-dessous avec $W = 6$:

j	1	2	3	4
u_j	40	15	20	10
w_j	4	2	3	1

Question 2. Comment peut-on transformer ce problème en un problème de recherche de plus court chemin ? Peut-on utiliser un algorithme de plus court chemin pour le résoudre ?

3 Couches convexes [5 points]

Etant donné un ensemble non vide E de n points dans le plan, on définit récursivement la suite des couches convexes de E :
- La première couche convexe C_1 est formée des points de l'enveloppe convexe de E
- Pour $i > 1$, soit E_i l'ensemble des points de E qui n'appartiennent pas aux couches convexes $C_1, \ldots, C_{i-1}$. Si E_i est vide alors la couche convexe C_i n'est pas définie, et sinon elle est formée des points de l'enveloppe convexe de E_i.
Question 1. Décrire un algorithme qui calcule la suite des couches convexes d'un ensemble de n points avec une complexité en $O(n^2)$. (Vous prouverez la complexité de l'algorithme que vous proposez.)
Question 2. On se propose maintenant de montrer que l'on peut réduire en temps linéaire un problème de tri à un problème de couches convexes. Soient A un tableau contenant n valeurs réelles non triées, et B un tableau contenant les mêmes valeurs que A, triées en ordre décroissant. Décrire un algorithme de complexité linéaire qui convertit A en un ensemble de points E, tel que E_i peut se traduire en temps linéaire en $B[i]$.
Question 3. En déduire une borne inférieure sur la complexité des algorithmes de calcul de la suite des couches convexes d'un ensemble de n points.