

Examen d'algorithmique - Types abstraits de données (Partie I)
Janvier 2003

Durée : 45 minutes.

Documents de cours, TD, TP autorisés. Les calculatrices ne sont pas autorisées.

Le sujet comporte 2 pages. Barème sur 20 donné à titre indicatif.

Exercice 1 : Algorithmes de tri (6 points)

1.1 Quelles sont les complexités théoriques (en nombre de transferts et en nombre de permutations) des algorithmes de tri à bulles et de tri par sélection/échange vus en cours?

1.2 Dérouler l'exécution de l'algorithme de tri à bulles sur le tableau suivant :

20	10	1	15	2	30	11
----	----	---	----	---	----	----

Quelles sont les complexités en nombre de transferts et en nombre de permutations du tri à bulles sur cet exemple précis?

1.3 Dérouler l'exécution de l'algorithme de tri par sélection/échange sur le même tableau que ci-dessus. Quelles sont les complexités en nombre de transferts et en nombre de permutations du tri par sélection/échange dans ce cas?

1.4 Les complexités effectives (sur cet exemple) des deux algorithmes de tri sont-elles en accord avec leurs complexités théoriques? Lequel de ces deux algorithmes de tri est donc le plus efficace?

Exercice 2 : Arbres binaires (14 points)

On suppose défini le TAD Arbre Binaire (ABin) vu en cours. Toutes les primitives de ce TAD sont donc utilisables pour cet exercice.

2.1 Écrire une fonction *entier resul-addition* (ABin A) retournant la somme de tous les éléments de l'arbre binaire d'entiers A. Quelle est la complexité de cette fonction?

2.2 Écrire une fonction *entier hauteur* (ABin A) retournant la hauteur d'un arbre binaire A. On conviendra que la hauteur d'un arbre vide vaut -1 .

Quelle est la complexité de cette fonction? Si cette fonction devait travailler sur un ABR, pourrait-on l'optimiser? Justifier vos réponses.

2.3 Écrire une fonction *entier minimum* (ABin A) retournant l'élément minimum d'un arbre binaire d'entiers A.

Quelle est la complexité de cette fonction *minimum*? Si la fonction *minimum* devait travailler sur un ABR, pourrait-on l'optimiser? Si oui, à quelle complexité pourrait-on s'attendre? Justifier vos réponses.

2.4 Indiquer les modifications à apporter à la fonction *minimum* pour obtenir une fonction *entier maximum* (ABin A) retournant l'élément maximum d'un arbre binaire d'entiers A.

2.5 Écrire une fonction *bool est-ABR* (ABin A) retournant vrai si l'arbre binaire A est ordonné selon le critère ABR et faux sinon. Quelle est la complexité de cette fonction?

Algorithmique
partie graphes et tables de hachage
1^{ère} session

Durée prévue de 45 mn. Tous documents autorisés.

Exercice 1 – Algorithme de recouvrement

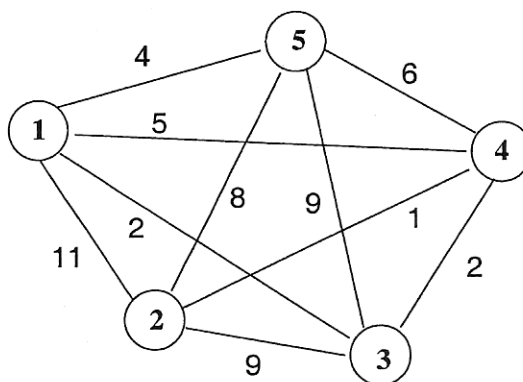


Figure 1: Un graphe

- 1- Appliquer l'algorithme de recouvrement au graphe de la figure 1, en partant du sommet 1.
- 2- Interpréter les résultats.

Exercice 2 – Tables de hachage

- 1- Donner une fonction de hachage coalescent avec réserve pour des numéros de sécurité sociale de la forme sexe année mois département ville rang où chaque élément est un nombre.
- 2- Donner un algorithme de suppression d'un élément dans une table de hachage coalescent avec réserve, avec résolution des collisions correspondant à des insertions à partir du début de la réserve. On suppose connus et disponibles les différents paramètres et fonctions permettant d'accéder à la table.

Exercice 3 – Question de TP

La déclaration en C du type correspondant à un sommet d'un graphe peut être :

```
typedef struct {
    char *nom;
    liste_t *cocyclePlus;
    liste_t *cocycleMoins;
} sommet_t;
```

- 1- Définir les fonctions nécessaires à la création d'une table de stockage des sommets d'un graphe accessibles par leur nom.