Université Bordeaux 1 Licence Informatique 2006/2007

INF102 Initiation à l'algorithmique

Devoir surveillé

Durée : 1h20. Aucun document autorisé. Les algorithmes qu'on vous demandera d'écrire dans les exercices suivants doivent utiliser le langage EXALGO défini en cours.

Exercice 1

1. Sans le reécrire, expliquer le principe de l'algorithme de tri par insertion.
2. Faites le tourner sur un tableau contenant la suite d'entier $4, 7, 2, 8, 6, 1, 5$ afin de trier le tableau en ordre croissant.
3. Donner un ordre de grandeur du nombre d'affectations et du nombre de comparaisons nécessaires au déroulement de cet algorithme pour le pire des cas, préciser quel est ce cas.
4. Même question pour le cas au "mieux"

Exercice 2

Soit `Mes` un tableau de `N` réels ($1 \leq N \leq NMAX$). En stockant par ordre croissant dans ce tableau une série de mesures on remarque que le tableau contient beaucoup de valeurs identiques qui sont contigües puisque le tableau est trié.

Exemple : avec $N = 16$

`Mes` =

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1.3	1.3	1.5	1.8	1.8	1.8	1.8	2.0	2.0	2.0	2.0	2.0	2.1	2.5	2.5	2.5

On souhaite représenter ce tableau de façon plus compacte en faisant figurer seulement les mesures dont les valeurs sont distinctes et en leur associant le nombre de fois où elles apparaissent dans le tableau `Mes`. Pour cela on utilise un tableau `MesCompact` dont les `M` éléments ($1 \leq M \leq KMAX$) sont des structures ayant deux champs :

- Un champ `mesure` contenant la valeur de la mesure
- Un champ `nb` contenant le nombre de fois que cette mesure figure dans le tableau `Mes`

Exemple : avec $M = 6$

`MesCompact` =

1	2	3	4	5	6
1.3	1.5	1.8	2.0	2.1	2.5
2	1	4	5	1	3

1. Les expressions suivantes ont-elles un sens ? Si oui donner leur type et leur valeur dans l'exemple précédent :

- `MesCompact[2]`
- `mesure[4]`
- `MesCompact.mesure[3]`
- `MesCompact[1].mesure`
- `nb.MesCompact[5]`
- `MesCompact[5].nb`

2. Ecrire une fonction `compacte` qui construit le tableau `MesCompact` à partir du tableau `Mes`. Les deux tableaux seront des paramètres de la fonction.
3. On suppose maintenant que le tableau `MesCompact` a été construit.
   - Ecrire une fonction `moyenne` qui calcule la moyenne de toutes les mesures à partir du tableau `MesCompact`.
   - Ecrire une fonction `freqmax` qui renvoie la valeur de la mesure qui apparait le plus souvent (en cas d'égalité, elle renvoie la plus petite mesure).

Exercice 3

On stocke des valeurs réelles strictement positives toutes distinctes dans un tableau `Vect` contenant `N` éléments ($1 \leq N \leq NMAX$).

Ce tableau est souvent modifié par des suppressions ou des ajouts d'éléments ; au lieu de faire des décalages, pour supprimer un élément on convient de le remplacer par 0 et pour ajouter un nouvel élément on l'ajoute à la fin du tableau s'il y a de la place. Il y a donc dans `Vect` des éléments « significatifs » : les réels strictement positifs, et des éléments « non significatifs » : les 0.

1. Ecrire une fonction `ajout-elt` qui ajoute dans le tableau `Vect` contenant déjà N éléments, l'élément dont la valeur est x (on supposera qu'il y a assez de place). Quelle est la complexité de cette fonction ?
2. Ecrire une fonction `sup-elem` qui « supprime » du tableau `Vect` un élément x, s'il est présent, en le rendant non significatif, sinon ne fait rien. Quelle est la complexité de cette fonction ?
3. Lorsque le nombre de 0 devient trop important, cela peut devenir gênant. Il est alors nécessaire de « tasser » le tableau `Vect` en supprimant les 0. Sans utiliser de tableau supplémentaire, écrire une fonction `tasser` qui supprime tous les 0 de `Vect`.

Exemple :

`Vect` $= (0, 0, 0, 11, 0, 23, 0, 0, 0, 17, 0, 0, 0, 0, 38, 0, 0, 0,)$ avec $N = 18$

devient

`Vect` $= (11, 23, 17, 38)$ avec $N = 4$