# Structures de données et algorithmes

Examen écrit, 24 août 2012

*Livres fermés. Durée maximale : 3h30.*

## Remarques

- Répondez à chaque question sur une feuille séparée, sur laquelle figurent votre nom et votre section. Soyez bref et concis, mais précis.
- Sauf mention explicite, toutes les complexités sont à décrire par rapport au temps d'exécution des opérations concernées. Soyez toujours le plus précis possible dans le choix de votre notation ($\Omega$, $O$ ou $\Theta$) et justifiez votre réponse.

## Question 1

Les algorithmes suivants prennent en entrée un tableau d'entiers $A$. Pour chacun des deux algorithmes, expliquez brièvement son effet sur le tableau $A$ en entrée et donnez les complexités au meilleur cas et au pire cas en fonction de la taille de l'entrée $n = A.length$. Justifiez votre réponse.

(a) ALGO-I($A$)

```
1  n = A.length
2  for i = 1 to n − 1
3      s = TRUE
4      for j = 1 to n − i
5          if A[j] < A[j + 1]
6              SWAP(A[j], A[j + 1])
7              s = FALSE
8      if s == TRUE
9          return
```

(b) ALGO-II($A$)

```
1  i = 1
2  j = A.length
3  while i < j
4      if A[i] > A[j]
5          SWAP(A[i], A[i + 1])
6          SWAP(A[i], A[j])
7          i = i + 1
8      else j = j − 1
```

## Question 2

(a) Qu'est-ce qu'une table de hachage ? Décrivez les opérations d'insertion et de suppression et donnez leur complexité.

(b) Soit une table de hachage de taille 11, et la fonction de hachage $h(k) = k \mod 11$. Les clés suivantes sont insérées dans la table dans cet ordre : $23, 34, 46, 57, 65, 76, 86, 97$. Choisissez deux méthodes de gestion des collisions parmi les trois ci-dessous et pour chacune d'elles, représentez la position des clés dans la table :
- Chaînage,
- Adressage ouvert avec sondage linéaire,
- Adressage ouvert avec double hachage, avec comme seconde fonction de hachage la fonction $g(k) = k \mod 5 + 1$.

(c) Soit un tableau d'entiers quelconques, on cherche à écrire un algorithme permettant d'afficher (dans un ordre quelconque) les doublons dans ce tableau. Par exemple, si le tableau est $[3, 2, 4, 3, 1, 2]$, l'algorithme affichera 3 et 2. Proposez un algorithme de complexité $O(n)$ pour résoudre ce problème.

# Question 3

(a) Qu'est-ce qu'un tas binaire (max) ? Comment implémente-t-on un tas dans un vecteur ?

(b) Ecrivez une fonction IS-HEAP prenant en argument un tableau $A$ et renvoyant TRUE si le tableau $A$ représente un tas binaire (max), FALSE sinon. Cette fonction devra être de complexité $O(n)$, où $n$ est la longueur du tableau $A$.

(c) Soit le tableau suivant $[30, 20, 7, 10, 5, 8, 2, 1]$. Représentez le tas généré à partir de ce tableau par la fonction BUILD-MAX-HEAP vue au cours.

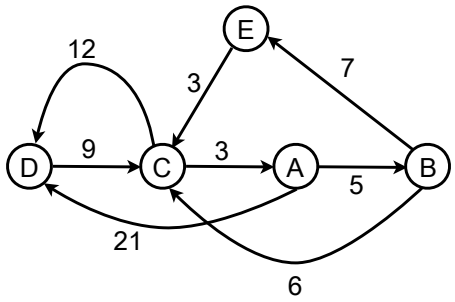
(d) Expliquez comment construire un tas binaire (max) à partir d'un arbre binaire de recherche en $O(n)$.

(e) Prouvez (par l'absurde) qu'il n'est pas possible de faire la transformation inverse en $O(n)$.

# Question 4

(a) Décrivez l'algorithme de Dijkstra, dans le formalisme de votre choix.

(b) Illustrez une exécution de cet algorithme sur le graphe pondéré ci-dessous à partir du nœud A.




(c) L'algorithme de Dijkstra continue-t-il à fonctionner lorsque certaines arêtes ont des poids négatifs (en supposant cependant qu'il n'y a pas de cycle de poids négatif) ? Justifiez votre réponse.
*Suggestion : exécutez l'algorithme sur le graphe précédent en remplaçant le poids de l'arête entre E et C par -3.*

# Question 5

Soit un tableau $A[1\,..\,n]$ de $n$ entiers, chacun pris dans l'intervalle $[1, k]$. On cherche à déterminer s'il existe un sous-ensemble des entiers qui somme exactement à $S/2$, où $S$ est la somme de tous les entiers de la liste.

(a) Ecrivez un algorithme efficace pour résoudre ce problème en utilisant la programmation dynamique.
*Suggestion : définissez une fonction $f(i, y)$ valant 1 s'il existe un sous-ensemble des $i$ premiers entiers qui somme exactement à $y$, 0 sinon. Il s'agit alors de calculer efficacement $f(n, S/2)$.*

(b) Analysez la complexité de cet algorithme en fonction de $n$ et de $k$.