Université Antilles–Guyane
UFR Sciences Exactes et Naturelles
Dépt Scientifique Interfacultaire

DEUG MIAS 1e année
option CGS
2e semestre 2004–2005

Mathématiques pour l'informatique — examen de juin 2005

Durée : 2 heures — Aucun document ni calculatrice autorisés.

Il sera accordé de l'importance au soin apporté à la rédaction et à la présentation des arguments et méthodes. Si vous pensez qu'il y a une erreur ou une imprécision dans l'énoncé, vous poursuivrez après avoir précisé la correction que vous jugez nécessaire.

Exercice 1. Questions de cours.

(a) Enumérer en ordre croissant (par rapport à la relation de domination, $f(n) = O(g(n))$ $(n \to \infty)$) les expressions suivantes :

$$n \ ; \ 2^n \ ; \ n \log n \ ; \ \log n \ ; \ \sqrt{n} \ ; \ e^n \ ; \ n^2 + \log n \ ;$$

$$5^{n/2} \ ; \ \log\log n \ ; \ (\log n)^2 \ ; \ n! \ ; \ 3^{n-3} \ .$$

Donner le nom de la classe d'équivalence grossière pour les expressions dont vous le connaissez.

(b) Expliquer le principe de la méthode de « diviser pour régner ».

Exercice 2. Déterminer un équivalent grossier (pour $n \to \infty$) de

$$f(n) = \sum_{k=1}^{n} k^2 \ln k \ , \quad g(n) = \sum_{k=1}^{n} \frac{\ln k}{k} \ , \quad h(n) = \sum_{k=1}^{n} \frac{\ln k}{k^2} \ .$$

(N'oubliez pas de vérifier les hypothèses des théorèmes utilisés.)

Tournez la page s.v.p. !

Exercice 3. On cherche le nombre maximal d'occurrences multiples d'un même élément dans un tableau `t` à $N \geq 1$ éléments.

(a) A défaut d'hypothèse supplémentaire, on cherche pour tout élément le nombre d'éléments identiques partout ailleurs dans le tableau. On considère l'algorithme suivant :

```
Algo_1 : nbre_maxi ← 0 ;  i ← 1
         Tant que i < N
              nbre_égaux ← 1 ; j ← i+1
              Tant que j ≤ N
                   Si  t[i] = t[j] : nbre_égaux ← nbre_égaux + 1
                   j ← j+1
              Fin Tant que
              Si  nbre_égaux > nbre_maxi : nbre_maxi ← nbre_égaux
              i ← i+1
         Fin Tant que
```

Pourquoi peut-on se limiter à $j > i$ pour la boucle imbriquée ?
Quelle est la classe de complexité (en fonction de N) de cet algorithme ?
(On négligera tout sauf les comparaisons d'éléments du tableau.)

(b) Donner un invariant de boucle pouvant être utile pour démontrer la correction partielle de l'algorithme ci-dessus. (Justifiez votre réponse, mais une preuve rigoureuse de l'invariance n'est pas demandée.)
Etant donné cet invariant, indiquer ce qui reste à faire pour prouver la correction de l'algorithme. (La preuve explicite n'est **pas** demandée.)

(c) On suppose maintenant le tableau trié, les éléments identiques étant donc à positions voisines.
Que fait l'algorithme suivant (rôle de `i` et `j`) ? Quelle est sa complexité ?

```
Algo_2 : nbre_maxi ← 1 ; i ← 1 ; j ← 1
         Tant que j < N
              j ← j+1
              Si  t[i] ≠ t[j]  faire
                   Si  j-i > nbre_maxi : nbre_maxi ← j-i
                   i ← j
              Fin faire
         Fin Tant que
```

(d) On considère à nouveau le cas (a) d'un tableau non trié.

i. Quelle est la classe de complexité des meilleurs algorithmes de tri que vous connaissez ? (Il suffit d'en nommer un.)

ii. Si on peut trier le tableau avec un algorithme de tri de complexité quasilinéaire, pouvez-vous alors proposer une méthode qui accomplit la tâche envisagée avec une complexité inférieure à celle de `Algo_1` du (a) ? Si oui, quelle est sa classe de complexité ?