

INFO2050 - Programmation avancée

Répétition 1: Pseudo-code et complexité

Jean-Michel BEGON

6 octobre 2017

Exercice 1

Que fait cette fonction ?

MYSTÈRE(A)

```
1  if  $A.length < 2$ 
2      return True
3  else
4      if  $A[1] == A[A.length]$ 
5          return MYSTÈRE( $A[2..A.length - 1]$ )
6      else
7          return False
```

Exercice 2

- (a) Ecrire le pseudo-code d'une fonction itérative permettant de déterminer la valeur minimale des éléments d'un tableau. Réécrire ensuite cette fonction de façon récursive.
- (b) Ecrire le pseudo-code d'une fonction récursive permettant de calculer les nombres de Motzkin :

$$M_N = \begin{cases} 1, & N = 0, N = 1 \\ \frac{3(n-1)M_{n-2} + (2n+1)M_{n-1}}{n+2}, & \forall N \in \mathbb{N}, N > 1 \end{cases}$$

Réécrire une version itérative de cette fonction.

- (c) Ecrire le pseudo-code d'une fonction récursive permettant de calculer le n -ième terme de la suite de Padovan

$$P_{n+3} = P_{n+1} + P_n \text{ avec } P_0 = P_1 = P_2 = 1$$

Réécrire ensuite cette fonction de façon itérative.

Exercice 3

- (a) L'algorithme A nécessite $10n^3$ opérations pour résoudre un problème. L'algorithme B résout le même problème en $1000n^2$ opérations. Quel est l'algorithme le plus rapide ?
- (b) L'algorithme A nécessite $32n \log_2 n$ opérations pour résoudre un problème. L'algorithme B résout le même problème en $3n^2$ opérations. Quel est l'algorithme le plus rapide ?

Exercice 4

Soit un algorithme dont le temps d'exécution pour $N = 1000, 2000, 3000$ et 4000 est respectivement de $5s, 20s, 45s$ et $80s$. Estimez le temps d'exécution pour $N = 5000$.

Exercice 5

Soit $f(n) = n$. Pour autant que ça soit possible, trouver une fonction $g(n)$ telle que

- $f(n) \in O(g(n))$ et $f(n) \notin \Omega(g(n))$
- $f(n) \notin O(g(n))$ et $f(n) \in \Omega(g(n))$
- $f(n) \in O(g(n))$ et $f(n) \in \Omega(g(n))$
- $f(n) \notin O(g(n))$ et $f(n) \notin \Omega(g(n))$

Exercice 6

- (a) Montrer que le temps d'exécution d'un algorithme est $\Theta(g(n))$ si et seulement si le temps d'exécution du pire cas est $O(g(n))$ et le temps d'exécution du meilleur cas est $\Omega(g(n))$.
- (b) Montrer que $5n^2 - 3n + 4$ est $\Theta(n^2)$.
- (c) Montrer que 2^{n+1} est $\Theta(2^n)$.
- (d) Expliquer pourquoi la phrase "Le temps d'exécution d'un algorithme A est au moins $O(n^2)$ " n'a aucun sens.

Exercice 7

Classer ces fonctions par ordre de complexité.

$n \log_2 n$	$\frac{4}{n}$	\sqrt{n}	2^{2^n}
$\log_2 \log_2 n$	$8n^3$	$8^{\ln n}$	$\frac{n}{2+n}$
$\log_2 n^7$	$5^{\ln \log_2 n}$	$(\log_2 n)^3$	$\frac{n}{\log_2(2+n)}$

Exercice 8

Pour chacun des pseudo-codes suivants, déterminer ce que fait l'algorithme, puis la complexité asymptotique en termes de n . (Soyez le plus précis possible sur les notations).

CODE1(n)

```
1 limit = n * n
2 sum = 0
3 for i = 1 to limit
4     sum = sum + 1
5 return sum
```

```

CODE2( $n$ )
1  $i = 1$ 
2  $limit = n * n * n$ 
3  $sum = 0$ 
4 while  $i < limit$ 
5      $sum = sum + 1$ 
6      $i = i * 2$ 
7 return  $sum$ 

```

```

CODE3( $a, b, c, n$ )
1 for  $i = 1$  to  $n$ 
2     for  $j = 1$  to  $n$ 
3          $a[i][j] = 0$ 
4         for  $k = 1$  to  $n$ 
5              $a[i][j] = a[i][j] + b[i][k] * c[k][j]$ 

```

Exercice 9

Soit un tableau A de n valeurs classées dans l'ordre croissant. On se propose de rechercher si une valeur b est présente dans ce tableau.

- Ecrire le pseudo-code d'un algorithme brutal pour rechercher la valeur b . Analyser sa complexité dans le meilleur cas et dans le pire cas.
- Proposer un algorithme dichotomique pour trouver la valeur b . Analyser sa complexité dans le meilleur cas et dans le pire cas.

Bonus

Bonus 1

Le projet Euler (<https://projecteuler.net/>) est une collection de problèmes informatiques demandant des implémentations efficaces. Le 4e problème est le suivant :

“Un nombre-palindrome indique la même valeur qu'on le lise de droite à gauche ou de gauche à droite.

Le plus grand palindrome résultant du produit de deux nombres à deux chiffres est $9009 = 91 \times 99$.

Trouver le plus grand palindrome résultant du produit de deux nombres à trois chiffres.”

Proposer un algorithme pour le résoudre.

Bonus 2

Soit un tableau $N \times N$ de booléens (0 ou 1). Proposer un algorithme pour trouver le plus grand sous-tableau contigu contenant uniquement des valeurs 1.

Exemple : Le tableau suivant contient un sous-tableau 4×4 contigu ne contenant que des 1.

```

1 0 1 1 1 0 0 0
0 0 0 1 0 1 0 0
0 0 1 1 1 0 0 0
0 0 1 1 1 0 1 0
0 0 1 1 1 1 1 1
0 1 0 1 1 1 1 0

```

0 1 0 1 1 1 1 0
0 0 0 1 1 1 1 0