

Algorithmique 4: Contrôle continu**- SUJET A -**

- Durée : 1 heure - Notes de cours autorisées uniquement -
Novembre 2022

Le barème est indicatif, il permet juste de comparer l'importance des exercices.

Toutes les réponses doivent être claires et justifiées.

Le sujet est à rendre avec votre copie.

Exercice 1.**Complexité d'algorithmes (3 points)**

Calculer la complexité de chacun des algorithmes suivants. L'instruction `<op_elem>` désigne n'importe quelle opération élémentaire, qui a complexité $O(1)$ par définition.

ALGO1(n) :

1. Pour $i = 0$ à $n - 1$:
2. Pour $j = 0$ à $n - 1$:
3. Pour $k = 0$ à j :
4. <op_elem>
5. Pour $i = 0$ à $n - 1$:
6. <op_elem>

ALGO2(n) :

1. <op_elem>
2. Tant que $n > 1$:
3. $n \leftarrow n/3$
4. <op_elem>
5. <op_elem>

ALGO3(n) :

1. Si $n \leq 1$: Renvoyer 0
2. <op_elem>
3. ALGO3($\lceil n/3 \rceil$)
4. ALGO3($\lceil n/3 \rceil$)
5. Pour $i = 0$ à n :
6. <op_elem>

Exercice 2.**Pile avec vidange (9 points)**

On veut implémenter une pile S possédant les opérations d'**EMPILER**, de **DÉPILER** et une opération **VIDANGE** qui supprime les $\lceil n/2 \rceil$ plus grands éléments de S , où n désigne le nombre d'éléments de S . On souhaite en plus que cette dernière opération s'exécute en temps amorti constant.

Pour réaliser cela, on considère que l'on a accès à une pile classique S pour laquelle les opérations **EMPILER**(S, x), de **DÉPILER**(S) sont fournies et empile l'élément x sur S et dépile S respectivement, le tout en temps constant.

On supposera tout au long de l'exercice que les valeurs entrées dans S sont toutes distinctes

1. Donner le contenu de la pile S , initialisée à $(23, 42, 1, 7, 18, 4)$, après les opérations **EMPILER**($S, 5$) et **VIDANGE**(S).
2. La **médiiane** d'un ensemble de n nombres distincts $X = \{x_1, \dots, x_n\}$ est le nombre x_i tel que exactement $\lceil n/2 \rceil$ nombres x_j vérifie $x_j < x_i$.
 - (a) Écrire un algorithme déterministe **MEDIANE** qui renvoie la médiane de l'ensemble X donné en paramètre. Évaluer la complexité de votre algorithme.
 - (b) Expliquer comment obtenir un algorithme probabiliste **MEDIANEPROBA** à partir de l'algorithme **QUICKSELECT** du cours. Préciser l'espérance de la complexité de cet algorithme.
3. Proposer alors un algorithme qui implémente **VIDANGE**(S) et qui fonctionne en temps $O(|S|)$ en moyenne. On pourra utiliser une pile auxiliaire (ou file, ou tableau...) si besoin. Pour la suite de l'exercice, on admettra l'existence d'un algorithme déterministe qui calcule la médiane de n nombres en temps $O(n)$, et donc que **VIDANGE**(S) peut fonctionner de manière déterministe en temps $O(|S|)$.
4. On part maintenant d'une liste S vide sur laquelle on souhaite effectuer une série de n opérations d'**EMPILER**, **DÉPILER** ou **VIDANGE**. Le but de la question est de montrer que le temps amorti moyen de ces opérations est en $O(1)$.
 - (a) **Par la méthode de l'agrégat.** Majorer le nombre d'éléments insérés au total dans S et conclure.
 - (b) **Par la méthode comptable.** On compte un coût amorti a_i de 3 si la i ème opération est **EMPILER** et 0 sinon. Montrer que pour tout $i = 1, \dots, n$, après la i ème étape, la somme des coûts amortis comptabilisés alors majore le nombre d'opérations élémentaires jusque-là : c'est-à-dire $\sum_{0 \leq j \leq i} c_j \leq \sum_{0 \leq j \leq i} a_j$ (où c_j est le nombre d'opérations élémentaires effectuées à la j ème étape). Conclure.
 - (c) **Par la méthode du potentiel.** On fixe un potentiel Φ à la pile S qui vaut après la i ème opération $3|S|$. Conclure.

Exercice 3.**Dominoes (8 points)**

Un **domino** est un couple d'entiers compris entre 1 et m . Pour deux dominos $D_1 = (i, j)$ et $D_2 = (k, l)$, on dit que D_2 se **juxtapose** à D_1 si $j = k$. Étant donné un ensemble X de n dominos, une suite D_1, \dots, D_p d'éléments de X est une **séquence** de X si D_{i+1} se juxtapose à D_i pour $i = 1, \dots, p - 1$. Une séquence de X est **complète** si elle contient tous les dominos de X , c'est-à-dire si elle taille n .

Par exemple, si $X = \{D_1 = (1, 3); D_2 = (4, 5); D_3 = (4, 1); D_4 = (5, 4); D_5 = (3, 4)\}$ alors X admet la séquence complète D_1, D_5, D_2, D_4, D_3 que l'on codera par le tableau $[1, 5, 2, 4, 3]$ des indices des dominos dans X .

Le but de l'exercice est d'écrire des algorithmes exhaustifs testant si un ensemble de dominos admet ou non une séquence complète. *Pour simplifier l'écriture des algorithmes, on supposera que tous les dominos de X sont distincts.*

1. Écrire un algorithme $\text{JUXTAPOS}(D, D')$ qui renvoie vrai si D' se juxtapose à D et faux sinon.
2. Proposer un algorithme de recherche exhaustive pour résoudre le problème, c'est-à-dire un algorithme $\text{DOMINOEXHAUS}(X)$ qui prend en entrée l'ensemble ordonné X de n dominos et retourne une séquence complète de X si X en admet une et faux sinon. On précisera bien quel type d'objets il est nécessaire d'énumérer pour cela. Donner la complexité de votre algorithme.
3. On veut maintenant écrire un algorithme de type *backtrack* pour ce problème.
 - (a) Dérouler un algorithme de ce type sur l'exemple $X = \{D_1 = (1, 3); D_2 = (4, 5); D_3 = (4, 1); D_4 = (5, 4); D_5 = (3, 4)\}$.
 - (b) Écrire l'algorithme $\text{DOMINOBACKT}(X)$ demandé.
 - (c) Donner sa complexité.
4. On s'intéresse maintenant aux séquences complètes D_1, D_2, \dots, D_n dites **circulaires** qui sont les séquences complètes pour lesquelles D_1 se juxtapose à D_1 . Expliquer comment modifier vos algorithmes DOMINOEXHAUS et DOMINOBACKT afin de tester si un ensemble X admet ou non une séquence complète circulaire.