sciences-centreimpression

De:	no-reply@www.usherbrooke.ca
Envoyé:	31 juillet 2015 15:24
À:	sciences-centreimpression
Objet:	COMMANDE EXAMENS- 7 août
Pièces jointes:	IFT436_Final_E2015.pdf

TYPE-EXAMEN	FINAL
SIGLE-COURS	IFT436
TITRE-COURS	Algorithmes et structures de données
PROFESSEUR	Richard St-Denis
DATE-HEURE	Vendredi 7 août à 9 h
AUTORISE-PAR	Richard St-Denis et André Mayers
NOMBRE-PAGES	12
NOMBRE-COPIE-PROF	44
IMPRESSION-QUESTIONNAIRE	Recto-verso plié broché en cahier
NOMBRE-FEUILLES-BLANCHES	
NOMBRE-PAPIER-GRAPHIQUE	
NOMBRE-CAHIERS	
CONSENTEMENT-AGES	1
REMARQUES	Impression en couleur dans un cahier s.v.p. (demande du Pr St-Denis).
	Richard.St-Denis@usherbrooke.ca
E-MAIL	
FIRST-NAME	
LAST-NAME	
NICK-NAME	
SPAMSHIELD	true

UNIVERSITÉ DE SHERBROOKE
DÉPARTEMENT D'INFORMATIQUE

IFT 436
Algorithmes et structures de données
EXAMEN FINAL [1]

Le vendredi 7 août 2015 de 9 h 00 à 12 h 00 Professeur : Richard St-Denis

- Toute documentation est permise.
- **Tout appareil électronique, incluant le téléphone cellulaire et l'ordinateur, est interdit.**
- Ne dégrafez pas ce questionnaire.
- Répondez dans les espaces prévus à cet effet.
- **Personne ne peut quitter la salle d'examen avant 9 h 15.**
- **Personne ne peut quitter son siège entre 11 h 50 et 12 h 00.**
- La correction est, entre autres, basée sur le fait que chacune de vos réponses soit :
  - claire, c'est-à-dire lisible et compréhensible pour le correcteur ;
  - précise, c'est-à-dire exacte ou sans erreur ;
  - complète, c'est-à-dire que toutes les étapes de résolution du problème sont présentes ;
  - concise, c'est-à-dire que la méthode de résolution est la plus courte possible.

Nom : ____________________ Prénom : ____________________

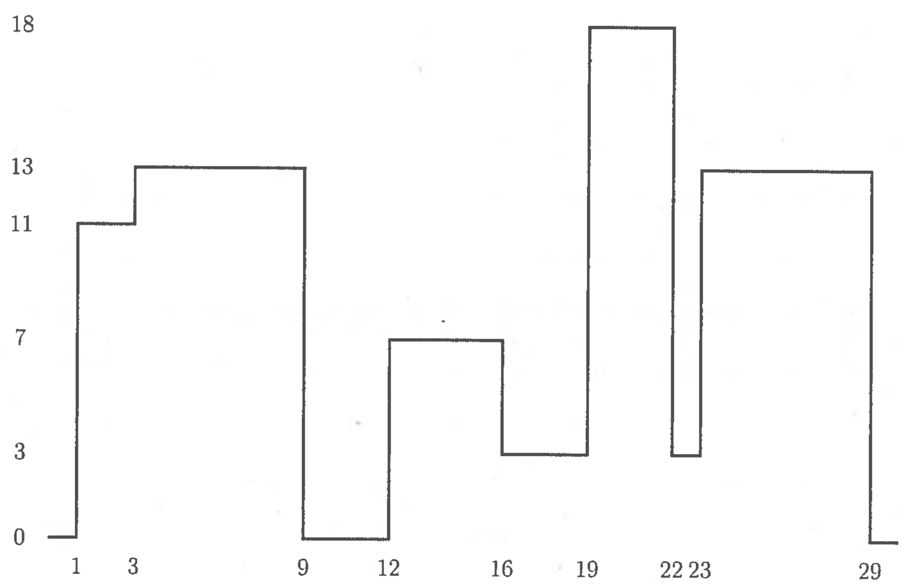
Signature : ____________________ Matricule : ____________________

Question	Barème
1	/8
2	/8
3	/8
4	/8
5	/8
total :	/40

1. Ce questionnaire comporte douze (12) pages.

1. (8 points) Dans le problème du calcul de la ligne d'horizon de bâtiments rectangulaires, il est suggéré de représenter la ligne d'horizon par une séquence d'entiers positifs : ceux de rang impair indiquent une position sur l'assise et ceux de rang pair indiquent une hauteur. La figure ci-dessous illustre la ligne d'horizon pour la séquence suivante de longueur 18 :

$$(1, \mathbf{11}, 3, \mathbf{13}, 9, \mathbf{0}, 12, \mathbf{7}, 16, \mathbf{3}, 19, \mathbf{18}, 22, \mathbf{3}, 23, \mathbf{13}, 29, \mathbf{0}).$$



Un autre représentation (ou structure de données) de la ligne d'horizon consiste à associer à chaque segment unitaire de l'assise une hauteur. Dans cette représentation, la séquence de hauteurs suivante (pour une assise de longueur 30) correspond à la même ligne d'horizon que celle définie en utilisant la première représentation :

$$\begin{aligned}(\quad & 0, 11, 11, 13, 13, 13, 13, 13, 13, 0, 0, 0, 7, 7, 7, 7, 3, 3, 3, 18, 18, 18, \\ & 3, 13, 13, 13, 13, 13, 13, 0 \quad).\end{aligned}$$

a) (5 points) Donnez, sous la forme de pseudocode **commenté**, un algorithme qui permet de passer de la deuxième représentation d'une ligne d'horizon à la première représentation de la même ligne d'horizon. Une réponse écrite dans un langage de programmation vaut zéro.

procedure *Transformation*(LH, l) **return** $\langle S, n \rangle$
{LH : ligne d'horizon selon la deuxième représentation}
{l : longueur de l'assise}
{S : ligne d'horizon selon la première représentation}
{n : longueur de la séquence S}

b) (3 points) Quelle est la complexité de votre algorithme en temps? Exprimez votre réponse dans la notation asymptotique la plus appropriée. Justifiez votre réponse.

2. (8 points) Le problème du parenthésage d'une expression arithmétique de façon à obtenir une valeur minimale ou maximale de l'expression, tel que présenté dans le cours, a été résolu en empruntant la technique de la programmation dynamique.

Tout d'abord, la solution consistait à attribuer des positions aux opérandes ou aux opérateurs de l'expression de façon à facilement identifier chacune de ses sous-expressions. Par exemple, pour l'expression

$$e = 1 + 2 \times 3 \times 4 + 5$$

le résultat de cette opération est :

$$e = {}_0 1_1 + {}_2 2_3 \times {}_4 3_5 \times {}_6 4_7 + {}_8 5_9.$$

Ainsi, $e(2,5)$ désigne la sous-expression 2×3.

Ensuite, une définition récursive de la valeur d'une solution optimale a été mise en évidence (le symbole $\odot$ représente l'opérateur d'addition $+$ ou de multiplication $\times$) :

$$\begin{aligned} T(i, i+1) &= n \text{ si } e(i, i+1) = n, \text{ un opérande entier} \\ T(i,j) &= \max_{i<k<j, e(k,k+1)=\odot} T(i,k) \odot T(k+1, j) \end{aligned}$$

Dans ces équations, i est pair. Dans la deuxième équation j est impair.

Enfin, une forme tabulaire, basée sur la définition récursive précédente, a été introduite. Il s'agissait, d'un triangle creux dont vous voyez ici un exemple après que les valeurs de chaque diagonale aient été calculées à partir de la définition récursive.

j	1	3	5	7	9
$i = 0$	1	3	9	36	81
$i = 2$		2	6	24	54
$i = 4$			3	12	27
$i = 6$				4	9
$i = 8$					5

a) (5 points) Explicitez, à l'aide de pseudocode **commenté**, le calcul de la valeur d'une solution optimale en mettant en évidence l'ordre dans lequel les valeurs des solutions optimales des sous-problèmes sont calculées.

(Répondez sur la page suivante)

b) (3 points) Quelle est la complexité de votre algorithme en temps et en espace ? Exprimez vos réponses dans la notation asymptotique la plus appropriée. Justifiez vos réponses.

3. (8 points) À l'automne votre voisin doit couvrir ses A arbustes, tous de différents volumes, pour les protéger du gel. Il se procure les A couvertures dont il a besoin pour couvrir tous ses arbustes chez un manufacturier qui, à cause de contraintes manufacturières, peut lui offrir le nombre suffisant de couvertures mais dans un nombre limité de tailles, disons $T \leq A$, à son choix. Bien entendu, une couverture fabriquée pour un volume donné peut couvrir un arbuste de plus petit volume. Comme le prix d'une couverture dépend du volume qu'elle peut couvrir (il n'est pas nécessaire de faire la différence entre un coût et un volume), votre voisin doit choisir les T tailles de couverture de façon à couvrir tous ses arbustes à moindre coût. Vous lui proposez donc de résoudre son problème en utilisant la programmation mathématique de la manière suivante.

   Supposons que les A arbustes sont ordonnés en ordre croissant par rapport à leur volume (l'arbuste 0 est celui de plus petit volume, l'arbuste $A-1$ est celui de plus grand volume) et que le coût pour couvrir l'arbuste i est dénoté par c_i $(0 \leq i \leq A-1)$. Ainsi, $c_0 < c_1 < \cdots < c_{A-1}$. La valeur optimale pour couvrir des arbustes, dont celui de plus grand volume est d'indice a, avec des couvertes de t différentes tailles est donnée par la définition récursive suivante :

$$f(a,t) = \begin{cases} c_a(a+1) & \text{si } t = 1 \\ \min_{a' \in \{t-2,\ldots,a-1\}} c_a(a-a') + f(a',t-1) & \text{si } t > 1 \end{cases} \tag{1}$$

   Dans la première équation de (1), il n'y a qu'une type de couverture $(t = 1)$. Il faut donc choisir le type de couverture qui permet de couvrir l'arbuste de plus grande taille, celui d'indice a et de coût c_a, et tous les autres arbustes non couverts, ceux d'indices $0, 1, \ldots, a-1$, et au même coût. Il y a donc $a+1$ arbustes à couvrir pour un coût total de $c_a(a+1)$.

   Dans la deuxième équation de (1), il y a plus d'un type de couverture $(t > 1)$. Il faut donc choisir un type de couverture, celui qui minimise l'expression $c_a(a-a') + f(a',t-1)$ pour $t-2 \leq a' \leq a-1$ [2], où

   - $c_a(a-a')$ est le coût total couvrir l'arbuste de plus grande taille, celui d'indice a et de coût c_a, et tous les autres arbustes non couverts d'indices $a'+1, \ldots, a-1$ et au même coût (donc $a-a'$ arbustes) ;
   - $f(a',t-1)$ est la valeur optimale pour couvrir des arbustes, dont celui de plus grand volume est d'indice a', avec des couvertes de $t-1$ différentes tailles.

2. $t-2$ et non pas 0, car plus le nombre de types de couverture augmente, plus il y en a de disponible pour couvrir les plus petits arbustes dont les coûts associés sont les plus petits.

Si $A = 10$, $T = 3$ et $\langle c_0, c_1, \ldots, c_9 \rangle = \langle 1, 4, 5, 7, 8, 12, 13, 18, 19, 21 \rangle$, alors une solution optimale est de couvrir les arbustes 9, 8 et 7 avec une couverture de coût 21, les arbustes 6 et 5 avec une couverture de coût 13 et les arbustes 4, 3, 2, 1 et 0 avec une couverture de coût 8, pour un coût total de 129.

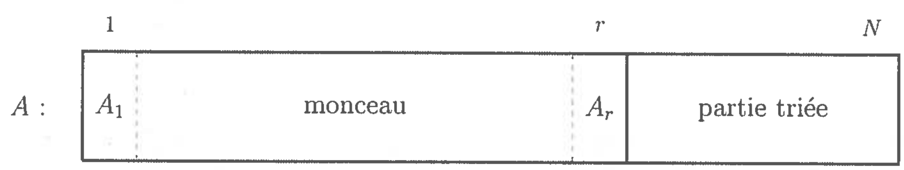
a) (2 points) Indiquez la forme tabulaire que vous choisissez pour ce problème de programmation dynamique.

b) (2 points) Explicitez, à l'aide de **pseudocode**, le calcul du coût d'une solution optimale en mettant en évidence l'ordre dans lequel les coûts des solutions optimales des sous-problèmes sont calculés (selon l'équation (1).

c) (2 points) Indiquez quelles informations doivent être conservées dans une structure de données de forme tabulaire, disons s, pendant le calcul du coût d'une solution optimale de façon à utiliser éventuellement ces informations pour produire une solution optimale.

d) (2 points) Donnez un algorithme sous forme de **pseudocode** qui génère le liste des tailles de couverture retenues et pour chaque taille les arbustes qu'elle permet de couvrir.

4. (8 points) Dans le tri en monceau, le monceau occupe les positions 1 à r dans le tableau A (voir la figure ci-dessous) à la r^{e} étape. La racine du monceau est à la position 1 et sa valeur est A_1. Ceci implique donc un échange de A_1 avec A_r (*swap*) et une opération de tamisage (*sift*) sur le monceau dans les positions 1 à $r-1$. La partie triée contient alors un élément de plus et r est décrémenté de 1.



Pour éviter l'échange de A_1 avec A_r, il serait intéressant de considérer un monceau pour lequel sa racine est en position r (donc à l'extrémité droite et de même pour les sous-monceaux). Deux difficultés surgissent :

— le calcul de *Left*(i) et de *Right*(i), respectivement les indices des racines des sous-arbres de gauche et de droite par rapport à la racine en position i ;

— le traitement effectué par l'opération de tamisage.

a) (2 points) Donnez deux formules une pour *Left*(i) et une autre pour *Right*(i) par rapport à cette nouvelle représentation. Dans le cas, où ce calcul serait impossible, indiquez-en la raison.

b) (2 point) Donnez le pseudocode **commenté** pour l'opération de tamisage par rapport à cette nouvelle représentation. Dans le cas, où ce traitement serait inefficace ou complexe, indiquez-en la raison.

c) (4 points) Pour déterminer la complexité de calcul de l'opération de tamisage, un raisonnement à partir de la **taille** d'un arbre quasi parfait a été retenu. Le même raisonnement s'applique également à un *poplar* puisqu'il s'agit d'un arbre binaire parfait (tout comme ses sous-arbres). Il se résumait comme suit :

> Appliquer l'opération de tamisage sur un arbre de taille $n = 2^{h+1} - 1$ (donc de hauteur h) consiste à comparer la valeur de sa racine avec les valeurs des racines de ses deux sous-arbres (de coût $\Theta(1)$) et, le cas échéant, d'effectuer un échange et d'appliquer l'opération de tamisage sur l'un ou l'autre des sous-arbres de taille $\lfloor \frac{n}{2} \rfloor = 2^h - 1$ (donc de hauteur $h - 1$).

Ceci conduit à l'équation de récurrence

$$T(n) = T(\frac{n}{2}) + \Theta(1)$$

qui a été résolue à l'aide du théorème maître.

Un autre raisonnement peut être fait à partir de la **hauteur** d'un *poplar* :

> Appliquer l'opération de tamisage sur un arbre de hauteur h consiste à comparer la valeur de sa racine avec les valeurs des racines de ses deux sous-arbres (de coût exactement 2) et, le cas échéant, d'effectuer un échange et d'appliquer l'opération de tamisage sur l'un ou l'autre des sous-arbres de hauteur $h - 1$.

Donnez l'équation de récurrence pour le coût de l'opération de tamisage, ainsi que la condition initiale, **en considérant cette fois-ci la hauteur des *poplars* et un coût de 2 (deux comparaisons) pour le traitement local sur les racines**, et trouvez la solution de l'équation de récurrence en utilisant la méthode appropriée.

Équation de récurrence : ______________________________

____

Condition initiale : ______________________________

____

Solution à l'équation de récurrence : ______________________________

____

____

____

____

____

____

5. (8 points) Répondez aux questions suivantes qui portent sur le tri progressif.

a) (3 points) Nous avons vu dans le cours que, pour trier une liste de 14 éléments, le tri *Poplar* considère successivement les configurations suivantes de *poplars*.

7-7
7-3-3
7-3-1-1
7-3-1
7-3
7-1-1
7-1
7
3-3
3-1-1
3-1
3
1-1
1

Chaque entier indique la taille d'un *poplar*. Par exemple, la configuration 7-3-1 représente une forêt de trois *poplars* : le premier de taille 7, le second de taille 3 et le dernier de taille 1.

En utilisant la même représentation, donnez la suite de configurations de tronçons considérée dans le tri progressif pour une liste de 14 éléments. Répondez dans les espaces ci-dessus.

b) (3 points) Représentez sous la forme d'une arborescence (dessinez un arbre) le tronçon de taille 15 obtenu à partir de la liste des nombres suivants : 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30.

c) (2 points) Quelle est l'utilité de la procédure *SemiTrinkel* dans le tri progressif.

FIN DE L'EXAMEN