Université d'Ottawa
Faculté de génie

École de science informatique
et de génie électrique

uOttawa

L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Electrical Engineering
and Computer Science

CSI2510/CSI2110 Structure de données et algorithmes

Examen final

Durée de l'examen: 3 heures **13 Décembre, 2011, 14:00**

Professeurs: J. Saboune & P. Flocchini **Page 1/13**

Nom de Famille :

Prénom:

Numéro d'étudiant:

Signature:

Livres fermés. Les calculatrices et tout appareil digital sont interdits.

S.V.P Ecrivez vos réponses dans l'espace reservé dans ce questionnaire.

Pages	notes de chaque page
PAGE 2	/5.5
PAGE 3	/4.5
PAGE 4	/6.5
PAGE 5	/2.5
PAGE 6	/4
PAGE 7	/3.5
PAGE 8	/3.5
PAGE 9	/1.5
PAGE 10	/3
PAGE 11	/4.5
PAGE 12	/2
PAGE 13	/4
TOTAL	/45

A la fin du temps d'examen:

- Tournez votre feuille d'examen.
- Restez silencieux et assis jusqu'à avoir la permission de sortir.

Question 1 [2 points] Quelle est la complexité temporelle (en notation asymptotique O) de ces deux algorithmes ? On considère n grand ($n > 100$)

Algorithm Hello(A)
Soit A un tableau de taille n.
for$i \leftarrow 1$ **to** n **do**
  for$j \leftarrow 10$ **to** n^2 **do**
    $A[i] \leftarrow j$

a) $O(\log n)$ b) $O(n)$ c) $O(n^2)$ d)$O(n^3)$ e)$O(n^4)$

Algorithm GoodLuck(n)
$i \leftarrow 1$; $j \leftarrow 1$;
while $i \leq n$ **do**
  { $j \leftarrow j + 3$;
  $i \leftarrow i * 2$;}

a) $O(\log n)$ b) $O(n)$ c) $O(n^2)$ d)$O(n^3)$ e)$O(n^4)$

Question 2 [2 points] Quelle la complexité temporelle au pire des cas pour la suppression d'une clé k dans les structures de données suivantes (en fonction de la taille n de la structure):

1. Tas-min (min-heap): *a)* $O(1)$ *b)* $O(\log n)$ *c)* $O(n)$ *d)* $O(n \log n)$
2. Arbre AVL: *a)* $O(1)$ *b)* $O(\log n)$ *c)* $O(n)$ *d)* $O(n^2)$
3. Table de hachage: *a)* $O(1)$ *b)* $O(\log n)$ *c)* $O(n)$ *d)* $O(n^2)$
4. Séquence triée (implémentée avec un tableau):
   a) $O(\log n)$ *b)* $O(n)$ *c)* $O(n \log n)$ *d)* $O(n^2)$

Question 3 [1.5 point] On veut trouver un algorithme pour transformer un arbre AVL de taille $n > 3$ en un tas-min (min-heap). Pour chacune des méthodes suivantes indiquez si elle accomplit le but cherché ou pas et dans le cas où elle est correcte indiquez sa complexité au pire des cas en notation asymptotique O.

1. Effectuez un parcours postfixe (post-order traversal) de l'arbre AVL et insérez les clés obtenues dans cet ordre dans un tableau qui représente un tas en commençant par l'indice 1:
   CORRECTE PAS CORRECTE Complexité:
2. Lire les clés de l'AVL dans un ordre quelconque et les insérez une par une dans un tas-min initialement vide en restructurant le tas après chaque étape:
   CORRECTE PAS CORRECTE Complexité: $O(n \log n)$
3. Lire les clés de l'AVL avec un ordre préfixe et effectuez une construction ascendante (bottom-up) du tas:
   CORRECTE PAS CORRECTE Complexité: $O(n)$

Question 4 [1.5 points] On veut utiliser le tri rapide sur place pour trier (dans un ordre croissant) les éléments du tableau suivant. Le pivot est choisi comme étant le dernier élément du tableau. Avant de faire les appels récursifs du tri on doit partitionner les éléments autour du pivot. Effectuez la première opération de partitionnement en montrant l'état du tableau après chaque échange nécessaire d'éléments:

- - Le tableau initial (18 étant le pivot):

1	2	3	4	10	15	25	20	1	30	5	18

- - Étape 1:

1	2	3	4	10	15	5	20	1	30	25	18

- - Étape 2:

1	2	3	4	10	15	5	1	20	30	25	18

- - Étape 3:

1	2	3	4	10	15	5	1	18	30	25	20

Question 5 [1.5 points] On veut construire un tas-max à partir de la séquence suivante $\{1, 3, 5, 7, 2, 4, 6, 8, 9, 10\}$ reprśentée par le tableau A ($A[1]$ contient 1, $A[2]$ contient 3, etc). Cette construction doit être effectuée selon la méthode de construction ascendante (bottom-up). Quel sera l'état du tableau après la fin de cette construction?

indice	1	2	3	4	5	6	7	8	9	10
valeur	10	9	6	8	3	4	5	1	7	2

Question 6 [1.5points] La table de hachage suivante est construite suivant une stratégie de double hachage ($h_j(k_i) = [h(k_i) + jd(k_i)] \bmod N$). La fonction de hachage primaire est donnée par $h(k_i) = k_i \bmod N$ et la fonction de hachage secondaire par $d_i(k_i) = k_i$ div N où div est la division d'entiers. La nombre maximal de cellules sondées est de $N = 13$. Insérez la clé 28 dans cette table et marquez les cellules sondées durant cette insertion:

i =	0	1	2	3	4	5	6	7	8	9	10	11	12
		27	15	14	17	available	6		28	35	18	25	

Question 7 [1.5points] La table de hachage suivante est construite suivant une stratégie de sondage quadratique avec une fonction de hachage $h(k_i) = (3k_i - 2) \bmod 11$. Recherchez la clé 23 dans cette table et indiquez les cellules sondées durant cette recherche.

i =	0	1	2	3	4	5	6	7	8	9	10
		1	5	9	2	available	10		7		

Question 8 [3 points] Utilisez l'algorithme de Dijkstra pour trouver l'arbre des plus courts chemins en partant du sommet A dans le graphe suivant représenté par sa liste d'adjacence (les nombres entre paranthèses représentent les poids des arêtes correspondantes).

$A \to B(4), C(5)$
$B \to A(4), D(3)$
$C \to A(5)D(1), E(3)$
$D \to B(3), C(1), E(1)$
$E \to C(3), D(1)$

Dessinez le graphe et remplissez le tableau suivant indiquant les distances (potentiellement mises à jour) après chaque ajout d'un nouveau sommet au nuage. La premiére ligne déjà remplie indique les distances initiales.

Prochain sommet	B	C	D	E	Arête ajoutée
A	4	5	∞	∞	—
B	4	5	7	∞	AB
C	4	5	6	8	AC
D	4	5	6	7	CD
E	4	5	6	7	DE

Question 9 [2 points] On utilise l'algorithme de Boyer-Moore pour trouver le pattern P dans la chaîne de caractères T. Indiquez, dans la table ci-dessous, les 7 premières comparaisons effectuées par l'algorithme. La première comparaison est déjà remplie .

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
T =	T	h	e		r	i	n	g		i	s		s	h	i	n	i	n	g
P =	s	h	i	n	i	n	g												

Comparaison #	i	j	$T[i]$	$P[j]$
1	6	6	n	g
2	7	6	g	g
3	6	5	n	n
4	5	4	i	i
5	4	3	r	n
6	11	6	-	g
7	18	6	g	g

Question 10 [0.5 point] Soit G un graphe avec n sommets et m arêtes. Quelle est la complexité temporelle de la recherche en profondeur (DFS) de G, si G est representé avec une matrice d'adjacence? $O(n^2)$

Question 11 [2 points] Vrai ou Faux: Confirmez ou infirmez les propositions suivantes

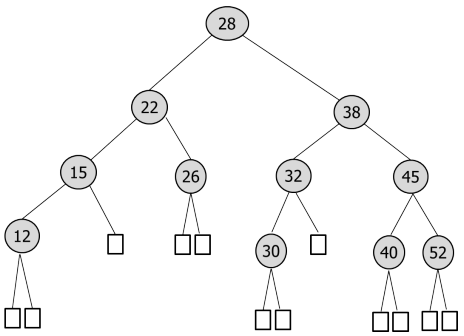
Chercher une clé dans une table de hachage est **toujours** plus rapide que chercher une clé dans un arbre binaire de recherche .	Vrai ☐	Faux ■
Dans un graphe où tous les poids sont égaux, le parcours en profondeur (DFS) et le parcours en largeur (BFS) vont visiter les sommets dans le même ordre (en partant du même sommet).	Vrai ☐	Faux ■
La recherche d'un sommet ou arête dans un graphe connexe peut toujours être effectuée en $O(m)$ (où m est le nombre d'arêtes), en choisissant une représentation adéquate du graphe.	Vrai ■	Faux ☐
Un arbre de recherche binaire avec les feuilles au même niveau est aussi arbre 2-4.	Vrai ■	Faux ☐

Question 12 [1.5 points] Soit G un graphe avec n sommets et $\frac{n^2}{3}$ arêtes représenté avec une liste d'adjacence. On veut utiliser l'algorithme de Prim pour trouver un arbre couvrant minimal (ACM) de G. Quelle structure serait plus bénéfique pour représenter la file de priorité utilisée dans cet algorithme: Un **Tas (Heap)** ou une **Séquence non-triée** ? Séquence non-triée

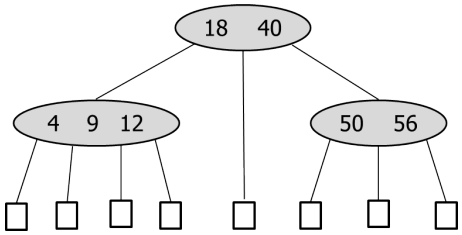
Quelle sera la complexité (en notation O fonction de n) de l'algorithme de Prim en utilisant la structure que vous venez de choisir comme file de priorité? $O(n^2)$

Question 13 [0.5 point] Lequel des deux parcours (en largeur BFS ou en profondeur DFS), appliqué à un tas en partant de la racine, retournerait les clés dans l'ordre avec lequel ils apparaiteraient dans une représentation par tableau du tas? BFS

Question 14 [2 points] Insérez la clé 30 dans l'arbre AVL suivant en utilisant l'algorithme vu en cours (identifiez x,y,z etc.). Dessinez l'arbre résultant après avoir appliqué toutes les modifications si nécessaires.



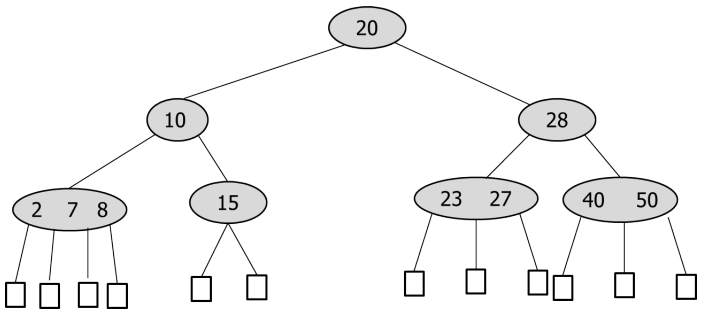
Question 15 [1 point] Confirmez ou infirmez les propositions concernant l'arbre suivant:



Cet arbre est un arbre de recherche généralisé Vrai ■ Faux □

Cet arbre est un arbre 2-4 Vrai □ Faux ■

Question 16 [2.5 points] Supprimez 35 de l'arbre 2-4 suivant. Dessinez l'arbre résultant après avoir appliqué toutes les modifications nécessaires.



Quelle est la complexité au pire des cas (en notation O) de la suppression d'une clé dans un arbre 2-4 avec n clés? $O(\log n)$

Question 17 [1.5 points] Effectuez la première "remontée de bulle" en vue d'un tri à bulles (en ordre croissant) de la séquence suivante. Décrivez l'état de la séquence après chaque comparaison:

10	24	18	39	7	31

10	24	18	39	7	31

10	18	24	39	7	31

10	18	24	39	7	31

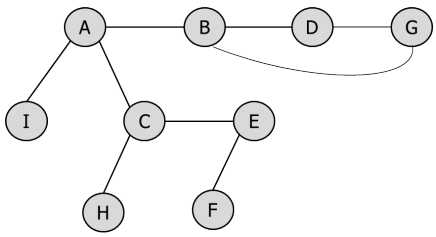
10	18	24	7	39	31

10	18	24	7	31	39

Question 18 [0.5 point] Qelle serait la complexité temporelle de l'algorithme de tri rapide (quicksort) si on choisit toujours le plus grand élément d'une séquence comme son pivot?

a) $O(log(n))$ b) $O(n))$ c) $O(nlog(n))$ d) $O(n^2))$ e) aucune

Question 19 [1.5 points] Confirmez ou infirmez les propositions concernant le graphe suivant:

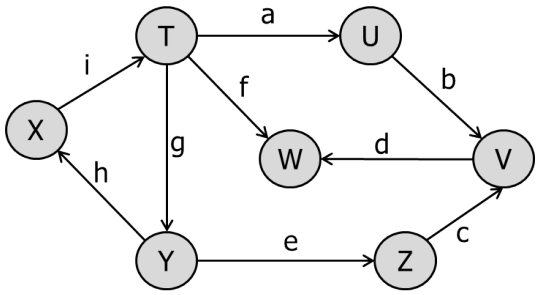


Ce graphe est connexe Vrai ■ Faux □

Ce graphe est acyclique Vrai □ Faux ■

On peut trouver 3 arbres couvrants différents pour ce graphe Vrai ■ Faux □

Question 20 [1.5 points] Remplissez la matrice d'adjacence, la liste d'adjacence, et la liste des arêtes correspondants au graphe orienté suivant. Listez les arêtes et les sommets suivant l'ordre alphabétique.



a) Matrice d'adjacence

	T	U	V	W	X	Y	Z
T	0	a	0	f	0	g	0
U	0	0	b	0	0	0	0
V	0	0	0	d	0	0	0
W	0	0	0	0	0	0	0
X	i	0	0	0	0	0	0
Y	0	0	0	0	h	0	e
Z	0	0	c	0	0	0	0

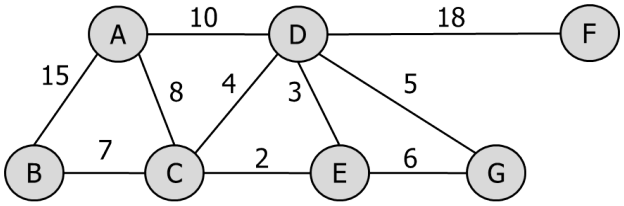
b) Liste d'adjacence

c) Liste d'arêtes

a	b	c	d	e	f	g	h	i

T	U	V	W	X	Y	Z

Question 21 [3 points] Trouvez un arbre couvrant minimal du graphe suivant en utilisant l'algorithme de Kruskal:



Remplissez la table suivante avec les arêtes choisies:

Arêtes choisies
CE
ED
DG
CB
CA
DF

Quel est le poids total de cet arbre? 43

Question 22 [3 points] On veut trier (en ordre croissant) les entiers ci-dessous en utilisant leur représentation en base 3 et l'algorithme de tri par base (radix-sort). Complétez la table suivante avec la représentation en base 3 des entiers avant d'appliquer l'algorithme de tri et aprés chaque étape de tri par paquets (bucket-sort).

Decimal	Base3	Tri 1	Tri 2	Tri 3	Tri 4
30	1010	1010	1001	1001	0002
2	0002	0120	2201	0002	0120
28	1001	1001	0002	1010	1001
15	0120	2201	1010	0120	1010
73	2201	0002	0120	2201	2201

Question 23 [1 point] Étant donné la séquence suivante:

5	25	36	38	41	49	50

Lequel de ces algorithmes va trier la séquence (en ordre croissant) avec le minimum d'opérations:

a) Tri à bulles b) Tri rapide c) Tri par fusion d) Tri par sélection

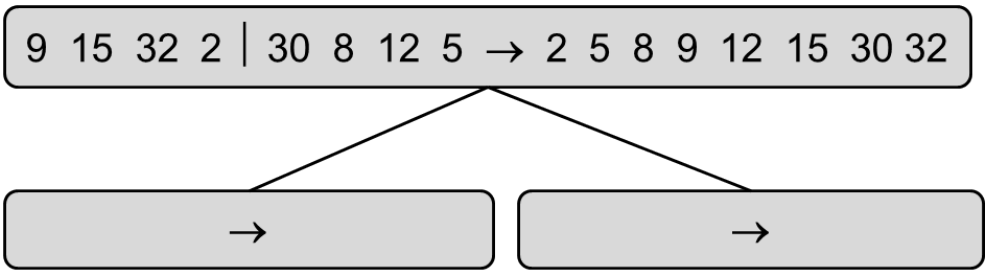
Quelle serait la complexité de cet algorithme pour cet exemple:

a) $O(log(n))$ b) $O(n))$ c) $O(nlog(n))$ d) $O(n^2))$ e) aucune

Question 24 [0.5 point] Étant donné un graphe orienté et connexe avec n sommets et m arêtes ; Laquelle de ces propositions est toujours valide:

a) $(n-1)/2 \leq m \leq (n) * (n-1)/2$
b) $(n-1) \leq m \leq (n) * (n-1)$
c) $(n-1)/2 \leq m \leq (n) * (n-1)$
d) $(n-1) \leq m \leq (n) * (n-1)/2$
e) aucune

Question 25 [2 points] Dessinez l'arbre complet d'éxécution du tri par fusion pour le tableau suivant:
N.B: Seulement les noeuds pour la première partition sont montrés



Question 26 [3 points] Pour un un graphe avec n sommets et m arêtes, quelle serait la complexité (en notation O fonction de n et/ou éventuellement m) des méthodes suivantes étant donné les représentations différentes du graphe:

	Matrice d'adjacence (implémentée avec un tableau (array) 2D)	Liste d'adjacence (implémentée avec un tableau de listes chaînées (linkedlists))	Liste d'arêtes (implémentée avec 2 tableaux (arrays))
`InsererSommet(v)`	$O(n^2)$	$O(n)$	$O(n)$
`SupprimerSommet(v)`	$O(n^2)$	$O(n+m)$	$O(n+m)$

Question 27 [1 point] La méthode `DTrav(G,v)` retourne une séquence S de sommets visités par un parcours en profondeur (DFS) d'un graphe non-orienté G en partant d'un sommet v. La méthode `SeqEq(S1,S2)` retourne un booléen indiquant si les deux séquences $S1$ et $S2$ contiennent les mêmes éléments. La méthode `G.Vertices` retourne une séquence contenant tous les sommets du graphe G.

Qu'indique l'algorithme `Unknown` décrit par le pseudo-code suivant?
N.B: Il faut trouver la finalité de cet algorithme au delà d'une description des opérations effectuées.

Algorithm `Unknown`

```
Pour un sommet aleatoire w de G
if SeqEq(DTrav(G,w),G.Vertices)
    return True
return False
```

retourne si un graphe est connexe ou pas