University of Ottawa
CSI 2531 – Examen de Mi-Session
Instructeur: Nathalie Japkowicz

Samedi le 3 Mars, 2001
13h00
Durée: 2 hs

Livres Fermés, Pas de Calculatrices
Version Francaise

Nom de Famille: ______________________

Prénom: ______________________

Numéro d'Etudiant: ______________________

Il y a 15 questions et un total de 15 points.

Cet examen doit contenir 17 pages,
en incluant cette page couverture.

1 – Enregistrements et Champs	/ 1
2 – Disques	/ 1
3 – Bandes Magnétiques	/ 1
4 – CD-ROM	/ 1
5 – Entrée/Sortie (I/O)	/ 1
6 – Mémoires Tampons	/ 1
7 – Compression Lempel-Ziv	/ 1
8 – Compression Huffman	/ 1
9 – Compression de Données	/ 1
10 – Récupération d'Éspace	/ 1
11 – Fragmentation	/ 1
12 – Triage et Rechèrche	/ 1
13 – Indexes Secondaires	/ 1
14 – Traitement Co-sequentiel	/ 1
15 – Traitement de Fichiers en C++	/ 1
Total	/ 15

1 Enregistrements et Champs — 1 point

Laquelle des propositions suivantes est **FAUSSE**:

A. Une clé primaire est une clé utilisée pour indentifier un enregistrement de manière unique.

B. Une clé peut être la combinaison d'un ou de plusieurs des champs d'un enregistrement.

C. Débuter un champ avec un indicateur de longueur ou séparer les champs avec des délimitateurs sont des méthodes utilisées pour les champs à longueur variable.

D. Les enregistrements à longueur fixe peuvent contenir des champs à longueur variable.

E. Si un fichier contient des enregistrements à longueur variable, le NER (numéro d'enregistrement relatif) peut être utilisé afin de calculer l'addresse (byte offset) d'un enregistrement.

... continued

2 Disques — 1 point

Soit un fichier contenant 16,000 enregistrements de 50 octets (bytes) chacun. Ce fichier doit être sauvegardé en mémoire secondaire. Trois disques sont disponibles et chacun d'entre eux peut sauvegarder 25,600 octets par piste et 4 pistes par cylindre. Néanmoins, chacun d'entre eux utilise une organisation différente:

- Disque 1:
  - Organisé par secteur
  - 128 octets par secteur
  - Aucun enregistrement n'a le droit d'utiliser plus d'un secteur (quoi que plusieurs enregistrements peuvent etre sauvegardés dans chaque secteur)
- Disque 2:
  - organisé par sous-blocs "Count-Data"
  - Facteur de Bloc = 50
  - Montant d'éspace additionnel par sous-bloc (utilisé par le sous-bloc count-data et l'éspace inter-bloc) = 60 octets
- Disque 3:
  - organisé par sous-bloc "Count-Data"
  - Facteur de Bloc = 10
  - Montant d'éspace additionnel par sous-bloc (utilisé par le sous-bloc count-data et l'espace inter-bloc)= 300 octets

Laquelle des propositions suivantes est **CORRECTE**?

A. Le Disque 1 nécéssite plus de cylindres que les Disques 2 et 3 pour sauvegarder le fichier.

B. Le Disque 1 nécéssite moins de cylindres que le Disque 2 mais plus de cylindres que le Disque 3 pour sauvegarder le fichier.

C. Le Disque 1 nécéssite plus de cylindres que le Disque 2 mais moins de cylindres que le Disque 3 pour sauvegarder le fichier.

D. Les trois disques nécéssitent autant de cylindres pour sauvegarder le fichier.

E. Le Disque 1 nécéssite moins de cylindres que les Disques 2 et 3 pour sauvegarder le fichier.

... continued

3 Bandes Magnétiques — 1 point

Veuillez considerer la description de bande magnétique suivante:

Densité = 3,000 bpi

Espace Inter-bloc = 0.2 inch

Vitesse = 200 ips

Veuillez supposer que l'on veuille sauvegarder un fichier contenant 1,000 enregistrements de 30 octets (bytes) sur cette bande magnétique en utilisant un facteur de bloc de 10. Quel est le **taux de transmission effective** de cette bande?

A. 1000 bpi

B. 200 Kilobytes/sec

C. 1 Megabyte/sec

D. 3000 bpi

E. 30 inches

... continued

4 CD-ROM — 1 point

Laquelle des propositions suivantes décrit un **avantage** des CD-ROMS?

A. Performance de recherche (seek) rapide.

B. Taux de transfert rapide.

C. Grande capacité de storage.

D. Organisation CLV (constant linear velocity).

E. Capabilité de Lecture et Ecriture.

... continued

5 Entrée/Sortie (I/O) — 1 point

Laquelle des propositions suivantes est **FAUSSE**:

A. Le processeur d'Entrée/Sortie (the I/O processor) est independent de l'Unité Logique (le CPU), ce qui permet aux opérations d'Entrée/Sortie et aux opérations internes d'etre effectuées en parallèle.

B. Le controlleur de disque est l'unité résponsable du control de l'opération de l'unité de disque.

C. Les tâches suivantes sont effectuées par le système d'exploitation/le géstionnaire de fichiers: la sauvegarde d'une table d'allocation des fichiers, la gestion de l'utilisation des mémoires tampons, la décision d'allouer un certain groupement (cluster) à un fichier.

D. S'il arrive que l'unité de disque est disponible pour l'écriture, la mémoire tampon du système d'Entrée/Sortie (the system I/O buffer) ne sera pas utilisée car les octets peuvent être écrits dirèctement sur le Disque.

E. Un groupement (cluster) consiste d'un ou de plusieurs secteurs contigus. L'utilisation de grands groupements peut améliorer le temps d'accès séquentiel, puisque de plus longs ségments de données peuvent être lus sans faire de rechèrche (seeking).

... continued

6 Mémoires Tampons — 1 point

Laquelle des propositions suivantes est **CORRECTE**?

A. Il existe beaucoup de types différents de mémoires tampons, dont la mémoire tampon de système qui s'occuppe de l'Entrée/Sortie (I/O) et la mémoire tampon de programmes qui s'occuppe des données d'un programme.

B. Lorsque l'on utilise une stratégie de "Buffer Pooling" (plusieures mémoires tampons sont utilisées en groupe) , il est conseillable d'utiliser la stratégie du MOST RECENTLY USED (utilisé le plus récemment) afin de séléctionner une mémoire tampon à utiliser. De cette facon, on peut être sûr que les mémoires tampons qui ont été utilisées le plus récemment ne restent pas en mémoire trop longtemps.

C. MÉMOIRE TAMPON DOUBLE, i.e. l'utilisation de 2 mémoires tampons, ralentit l'opération d'Entrée/Sortie lorsqu'un programme doit ET lire du disque ET écrire sur le disque pendant son execution.

D. L'utilisation de mémoires tampons rend les opérations d'Entrée/Sortie plus efficaces seulement lorsque l'excecution en parallèle du système d'Entrée/Sortie et de l'Unité Logique (CPU) NE sont PAS permises.

E. Lorsqu'un programme fait une erreur fatale, le contenu des mémoires tampons que le programme utilisaient avant cette erreur est toujours sauvegardé sur storage secondaire.

... continued

7 Compression Lempel-Ziv — 1 point

Veuillez considérer la chaine de caractères suivante:

```
ababcaabcabc
```

Si on suppose que 8 bits sont normalement nécéssaires pour encoder chaque caractère, combien de bits sont épargnées par un codage Lempel-Ziv de ce message?

A. 0 bits

B. entre 1 et 20 bits (inclus)

C. entre 21 et 40 bits (inclus)

D. entre 41 et 60 bits (inclus)

E. plus de 60 bits

8 Compression Huffman — 1 point

Veuillez considerer un fichier contenant les lettres suivantes accompagnées de leurs fréquences:

lettre:	A	B	C	D	E
fréquence:	40	20	15	14	11

Veuillez choisir le seul CODE HUFFMAN VALIDE pour ce fichier:

A.

lettre:	A	B	C	D	E
CODE:	0	10	110	1111	1110

B.

lettre:	A	B	C	D	E
CODE:	0	111	110	101	100

C.

lettre:	A	B	C	D	E
CODE:	11	10	01	001	000

D.

lettre:	A	B	C	D	E
CODE:	0	11	10	01	00

E.

lettre:	A	B	C	D	E
CODE:	0	111	101	110	100

... continued

9 Compression de Données — 1 point

Laquelle des propositions suivantes est **CORRECTE**:

A. La compression est rarement utilisée puisque le temps nécéssaire à la compression ne justifie pas le montant d'éspace épargné grâce à cette compression.

B. Il est important de considerer le compromis (tradeoff) temps/éspace habituel lorsque l'on utilise la compression de données: bien que la compression de données nous permet d'utiliser moins de storage sur le disque, elle augmente le temps nécéssaire pour transmettre le fichier sur un réseau tel que l'internet.

C. RUN-LENGTH ENCODING, la méthode de compression qui consiste à abrèger les valeurs contigues répétées, est plus effective pour la compression de texte que pour celle d'images.

D. Les stratègies de codage Lempel-Ziv et Huffman sont des téchniques de compression qui épargnent beaucoup d'éspace et qui sont routinement utilisées dans les systèmes UNIX.

E. Les stratègies de codage Lempel-Ziv et Huffman sont toutes deux capables d'atteindre le même nombre de bits moyen pour l'encodage de chaque symbole. Néanmoins, le codage Lempel-Ziv demande aussi la connaissance des probabilités de présence de chaque symbole.

... continued

10 Récupération d'Éspace — 1 point

Veuillez considerer le fichier suivant contenant des enregistrements à longueur variable et à champs de longueur variable:

```
26Smith|James|Dec|17|1973|
38Dupont-Sur-Seine|Michel|Feb|22|1970|
21Elk|Ann|Mar|6|1976|
31Shelling|Anthony|May|29|1969|
```

La longueur de chaque enregistrement est indiquée par un indicateur de longueur au début de chaque enregistrement. Les champs sont séparés par le délimitateur: '|'.

Le fichier contient 4 enregistrements: R1 de longueur 26, R2 de longueur 38, R3 de longueur 21 et R4 de longueur 31.

(Note: dans le fichier réel, il n'y a pas de retour-chariot après chaque enregistrement. Nous les avons insérés, ici, pour clarifier la lecture).

On utilise une liste d'enregistrements disponibles (AVAIL LIST) afin de récupérer l'éspace des enregistrements effacés. Veuillez supposer qu'une stratégie WORST-FIT (l'appariement le pire) est utilisée afin de réduire la fragmentation interne. Veuillez aussi supposer que AVAIL LIST est initiallement vide.

Le fichier juste décrit a été modifié de la manière suivante (et dans l'ordre indiqué):

- enregistrement R3 est effacé,
- enregistrement R5 de taille 25 est ajouté,
- enregistrement R4 est effacé,
- enregistrement R6 de taille 17 est ajouté.

Quel est le contenu de AVAIL LIST (ne considérant que la taille des blocs qu'elle inclus) une fois que ces modifications ont été pérformées?

A. 31, 21

B. 26, 38, 25, 17

C. 31

D. 21, 14

E. 26, 38, 17, 31, 25

... continued

11 Fragmentation — 1 point

Laquelle des propositions suivantes est **FAUSSE**?

A. L'éspace gaspillé a l'intérieur d'un enregistrement s'appelle la FRAGMENTATION INTERNE.

B. L'effacement d'enregistrements dans des fichiers contenant des enregistrements à longueur variable peut causer de la FRAGMENTATION EXTERNE.

C. LA COMPACTION DE STORAGE peut être utilisée pour combattre la fragmentation externe.

D. LA FRAGMENTATION peut être traitée de facon dynamique en récupèrant les éspaces effacés lorsque des enregistrements sont ajoutés.

E. LA FRAGMENTATION EXTERNE est souvent PIRE lorsque les enregistrements effacés sont de longueur fixe plutot que de longueur variable.

... continued

12 Triage et Rechèrche — 1 point

Laquelle des propositions suivantes est **CORRECTE**?

A. L'un des avantages du TRIAGE PAR CLE (KEYSORTING) est qu'il nécéssite moins de mémoire primaire que le triage interne d'un fichier.

B. LE TRIAGE INTERNE d'un fichier réferre à n'importe quelle méthode de triage qui prend place en mémoire primaire. Ceci inclut la situation dans laquelle le fichier ne tient pas complètement en mémoire primaire.

C. LA RECHERCHE SEQUENTIELLE dans un fichier prend moins de temps que la RECHERCHE BINAIRE dans ce même fichier car la lecture séquentielle d'un fichier est plus rapide que l'accès direct (seeking) dans ce fichier.

D. LA RECHERCHE BINAIRE dans un fichier de n enregistrements (pour un n de valeur arbitraire) prend, au pire, n accès au disque.

E. LE TRIAGE INTERNE d'un fichier de n enregistrements (pour un n de valeur arbitraire) demande, au moins, $n \times \log n$ accès au storage secondaire.

13 Indexes Secondaires — 1 point

Veuillez considerer un fichier de disques (musicaux) semblable a celui que nous avons discuté en classe. La structure suivante représente un **indexe de compositeurs**, qui est un **indexe secondaire** utilisant des **listes inversées**:

Indexe Secondaire:

0	BEETHOVEN	2
1	DVORAK	3
2	PROKOFIEV	5

Liste de References des Clés Primaires:

0	DG139201	4
1	LON2312	-1
2	ANG3795	0
3	COL31809	-1
4	RCA2626	-1
5	ANG36193	1

Veuillez supposer que l'enregistrement de données suivant:

```
DG139201|Symphony No.9|BEETHOVEN|
```

est effacé du fichier de donnees et que l'indexe de compositeurs est mis à jour de manière appropriée.

(continuer sur la page suivante...)

APRÈS CET EFFACEMENT, l'indexe de compositeurs devient:

A.

0	DVORAK	3
1	PROKOFIEV	5

0	********	*
1	LON2312	-1
2	ANG3795	4
3	COL31809	-1
4	RCA2626	-1
5	ANG36193	1

B.

0	BEETHOVEN	2
1	DVORAK	3
2	PROKOFIEV	5

0	********	*
1	LON2312	-1
2	ANG3795	0
3	COL31809	-1
4	RCA2626	-1
5	ANG36193	1

C.

0	BEETHOVEN	2
1	DVORAK	3
2	PROKOFIEV	5

0	********	*
1	LON2312	-1
2	ANG3795	-1
3	COL31809	-1
4	RCA2626	-1
5	ANG36193	1

D.

0	BEETHOVEN	2
1	DVORAK	3
2	PROKOFIEV	5

0	********	*
1	LON2312	-1
2	ANG3795	4
3	COL31809	-1
4	RCA2626	-1
5	ANG36193	1

E.

0	BEETHOVEN	1
1	DVORAK	3
2	PROKOFIEV	5

0	********	*
1	LON2312	-1
2	ANG3795	4
3	COL31809	-1
4	RCA2626	-1
5	ANG36193	1

... continued

14 Traitement Co-sequentiel — 1 point

Combien d'**opérations de lecture** et de **comparaisons de clés** sont nécéssaires afin d'**apparier (match)** les deux listes de lettres suivantes, si on suppose que les deux listes sont sauvegardées sur disque:

```
List1: B, E, F, K, L, P, Z
List2: A, B, F, G
```

Notes Importantes: Veuillez seulement compter les **opérations de lecture** dans lesquelles une clé (lettre) est lue; Un éssai de lecture de clé qui détécte une "fin de fichier" ne doit pas être compté comme une opération de lecture.

A. nombre d'opérations de lecture = 8 nombre de comparaisons de clés = 5

B. nombre d'opérations de lecture = 8 nombre de comparaisons de clés = 8

C. nombre d'opérations de lecture = 11 nombre de comparaisons de clés = 8

D. nombre d'opérations de lecture = 11 nombre de comparaisons de clés = 5

E. nombre d'opérations de lecture = 11 nombre de comparaisons de clés = 28

... continued

15 Traitement de Fichiers en C++ — 1 point

Veuillez considérer, ci-dessous, le ségment d'un programme en C++:

```
#include <fstream.h>
int main() {
  char ch = 'x';
  fstream f;
  f.open("myfile.txt",ios::in);
  .
  .
  .
}
```

Laquelle des opérations de traitement de fichier suivantes est GARANTIE DE CAUSER UNE ERREUR si elle est executée par le programme?

A. `f << c;`

B. `f >> c;`

C. `f.seekg(0,ios::beg);`

D. `f.seekg(10;ios::cur);`

E. `f.close();`