18 octobre 2004

UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS
DÉPARTEMENT D'INFORMATIQUE ET D'INGÉNIERIE

STRUCTURES DES DONNÉES ET ALGORITHMES
INF4393

EXAMEN INTRA

Directives:

1. C'est un examen à livre ouvert.
2. Répondre directement sur ce formulaire.
3. L'examen dure trois heures.
4. Le barème est montré au début de chaque question.
5. Les questions peuvent être posées seulement durant le 30 premières minutes de l'examen.

Nom, prénom:_____

Code permanent:_____

Signature:_____

Numéro de formulaire:_____

Ex. #	1	2	3	4	5	6	7	8	9	Σ
Note										
sur	10	9	7	12	15	9	16	12	11	100

Ex. #1 (10 pts)

Soit la déclaration en langage C:

int Tab [3] [2] [5];

Supposons que sizeof int = 4 (c'est-à-dire que chaque élément entier de ce tableau occupe 4 octets), et que le tableau a été alloué à partir de l'adresse de base = 2000.

a) Quelle est la taille de la zone mémoire occupée par le tableau Tab?

b) Quelle est la taille de la zone mémoire occupée par l'élément Tab [1] de ce tableau?

c) Quelle est l'adresse de l'élément Tab [2, 1, 2] de ce tableau?

d) Quelle est l'adresse de l'élément Tab [2, 0] de ce tableau?]

e) Quel élément du tableau Tab occupe la case mémoire à l'adresse 2074?]

Ex. #2 (9 pts)

Répondre par **Oui** ou **Non**:

a) O(n) = 3n-1 ?

b) O(3n-1) = n ?

c) O($\sqrt[2]{n}$) = n ?

d) O(n) = $\sqrt[2]{n}$?

e) Θ(n) = 3n-1 ?

f) Θ(3n-1) = n ?

g) Θ($\sqrt[2]{n}$) = n ?

h) Θ(n) = $\sqrt[2]{n}$?

i) Θ($\log_2 n$) = $\log_3 n$?

Ex. #3 (7 pts)

Afin de prouver pour la fonction $f(n) = 2n^3 - 60n$, que nous avons $f(n) = \Theta(n^3)$, il faut choisir les constantes c_1, c_2 et n_0. Encercler la reponse correcte parmi les suivantes:

a) $c_1 = 3/2$, $c_2 = 2$, $n_0 = 10$,
b) $c_1 = 3/2$, $c_2 = 3$, $n_0 = 11$,
c) $c_1 = 23/12$, $c_2 = 2$, $n_0 = 28$,
d) $c_1 = 23/12$, $c_2 = 3/2$, $n_0 = 27$,
e) **a)** et **b)**,
f) **a)**, **b)** et **c)**,
g) **b)**, **c)** et **d)**,
h) **c)** et **d)**,
i) toutes les reponses **a)**, **b)**, **c)** et **d)**
j) aucune parmi ces reponses

Ex. #4 (12 pts)

Le tableau tab doit être trié par un algorithme de tri par bulles:

```
#include <stdio.h>
#define Max 11

TriParBoules(int a[ ], int N)
{
   int i, j, temp;

   for ( i=N; i>=1; i--)
      for (j=2; j<=i; j++)
         if (a[j-1] > a[j]) {
            temp = a[j-1];
            a[j-1] = a[j];
            a[j] = temp;
         }
}

main()
{
    int tab[Max] =
        {0, 41, 46, 17, 77, 5, 32, 84, 41, 55, 85};
    TriParBoules (tab, 10);
}
```

a) (6 pts) Quel est le nombre total de comparaisons d'éléments du tableau `tab` faites par cet algorithme?

b) (6 pts) Quel est le nombre total d'échanges?

Ex. #5 (15 pts)

Soit un programme contenant un appel d'une fonction récursive tri_rapide:

```
#include <stdio.h>
#define MAX 10

echanger (int a[ ], int i, int j)
{   int temp;

    temp = a[i]; a[i] = a[j]; a[j] = temp;
}

afficher_tableau (int a[ ], int nbre)
{   int i;

    for (i = 1; i <= nbre; i++)
        printf ("%d  ", a[i]);
    printf ("\n");
}

tri_rapide (int a[ ], int gauche, int droite)
{   int val, i, j;

    if (droite>gauche) {
        val = a[droite]; i = gauche-1; j = droite;
        for ( ; ; ) {

            while (a[++i] < val) ;
            while (a[--j] > val) ;
            if (i>=j) break;
            echanger ( a, i, j);
        }
        echanger ( a, i, droite);
        afficher_tableau (a, MAX-1);
        tri_rapide (a, gauche, i-1);
        tri_rapide (a, i+1, droite);
    }
}

main ()
{   int i, tab [MAX] = {0, 2, 4, 6, 9, 8, 1, 3, 7, 5};

    tri_rapide (tab, 1, MAX-1);
}
```

a) (7 pts) Donner l'arbre de récursivité de l'appel principal `tri_rapide(tab, 1, MAX-1)`.

b) (5 pts) Qu'est-ce qui est affiché par ce programme?

c) (3 pts) Quelle est le nombre d'échanges effectués par ce programme?

Ex. 6 (9 pts)

Soit la fonction récursive f définie comme suit:

Cette fonction peut être calculée par le programme ci-dessous:

```
#include <stdio.h>
int f (int n)
{int i, j;
    if (n<=3) {
        printf ("n=%d\n", n);
        return (n);
    } else /* n>3 */ {
        i = f((n+2)/2);
        printf ("apres i; n=%d\n", n);
        j = f ((n+3)/2);
        printf ("apres j; n=%d\n", n);
        return (i + 2*j);
    }
}
 main ()
{
    printf("f(11) = %d\n", f(11));
}
```

Donner l'arbre de récursivité de l'appel `f(11)`.

Ex. #7 (18 pts)

Pour le programme de l'**ex. #6**

a) (3 pts) Donner la valeur retournée par l'appel `f(11)` du programme principal.

b) (5 pts) Donner la sortie affichée par ce programme.

c) (2 pts) Donner le nombre total d'appels de la fonction `f`.

d) (3 pts) En supposant que le programme main de l'ex. #4 contient l'appel de `f(25)` à la place de `f(11)`, donner le nombre total d'appels de la fonction `f`.

e) (3 pts) Dans le cas mentionné dans l'ex. #5 d), donner la valeur retournée par l'appel `f(25)` du programme principal.

Ex. #8 (12 pts)

Soit une suite suivante de déclarations en langage C:

```
typedef struct{
    char sigle [7];
    double moyenne;
} cours;

typedef struct{
    char matricule [12];
    cours notes [30];
} etudiant;

etudiant groupe [100];
```

Supposons que sizeof double = 4 avec l'alignement de 4 et sizeof char = 1. La variable groupe a été allouée à partir de l'adresse de base = 1000.

a) Quelle est la taille de la zone mémoire occupée par la variable groupe?

b) Quelle est la taille de la zone mémoire occupée par la donnée groupe[5]?

c) Quelle est l'adresse de la donnée groupe[5]?

d) Quelle est l'adresse de la donnée groupe[5].notes[10]?]

e) Quelle est la taille de la zone mémoire occupée par la donnée groupe[5].notes[10]?]

f) Quelles composantes de données occupent la case mémoire à l'adresse 2,503?]

Ex. #9 (11 pts)

Soit la fonction de la recherche dichotomique vue pendant le cours:

```
int FindInSortedArray (int key, int array[], int n)
{
    int lh, rh, mid, val;

    lh = 0;
    rh = n - 1;

    while (lh <= rh) {
        mid = (lh + rh) / 2;
        val = array[mid];
        if (key == val) return mid;
        if (key < val) {
            rh = mid - 1;
        } else {
            lh = mid + 1;
        }
    }
    return (-1);
}
```

Combien de fois est exécutée la boucle while de la fonction FindInSortedArray, si elle

a) (3 pts) est appelée pour le tableau array = {0, 3, 6, 8, 10, 12, 14, 15, 18}, pour la recherche de la valeur 14?

b) (3 pts) est appelée pour le tableau array = {0, 3, 6, 8, 10, 12, 14, 15, 18}, pour la recherche de la valeur 17?

c) (5 pts) est appelée pour le tableau array contenant 200 éléments (c-à-d a[0], a[1], ... a[199]), pour la recherche de la valeur se trouvant dans array[23]?