Mid-Term Exam of ***Data Structures*** (Fall 2017)
Department of Information Management and Finance
National Chiao Tung University

> The answers should be brief and precise, including C/Java-like pseudo codes.
> The input and output of any algorithm should be specified explicitly.

PART 1: True or False (Explain the reason if your answer is FALSE.) (10 pts.)

1. (T) $4096 = \Theta(1)$
2. (T) $n^3 + \log n = \Theta(n^3)$
3. (F) $n^2 \log n = \Omega(n^3)$
4. (T) $\sum_{i=0}^{n} i^2 = O(n^3)$
5. (F) Notation big-O aims at measuring the lower bound of a function, while Ω focuses on the upper bound.
6. (T) In C, the memory locations allocated for a static 2-D memory are consecutive; whereas, those of a dynamic allocation are not.
7. (F) The "undo" (Ctrl-Z) capability of many software applications could be implemented by using a queue.
8. (T) In C language, `&data[i][j]` is equivalent to `data[i]+j` and `*(data+i)+j`.
9. (F) If there is an algorithm A solving P, which has a lower bound $\Omega(g(n))$, in time $O(g(n))$ in the worst case, then A is an optimal algorithm for P. Thus, the optimal algorithm of P is unique (which is exactly A).

PART 2: Answer the following questions:

1. Determine the time complexity of the following pieces of programs using the tightest big-O notation.

(a) (3%)
```
for (x=1; x<=n; x*=2)
    for (y=1; y<=x; y++)
        z++;
```
(c) (3%)
```
for (x=1; x<=n; x*=2)
    for (y=1; y<=x; y*=2)
        z++;
```

2. Suppose that n integers are stored in 1D array A. We would like to search target x in A and expect index k if $A[k] == x$, or -1 otherwise (x cannot be found in A).
   (a) (2 pts.) Design a *linear search* algorithm to search x in A.
   (b) (5 pts.) Design a *binary search* algorithm to search x in B where B is another array consisting of all data in A in the non-decreasing order.
   (c) (3 pts.) When the number of searches comes to k, compare the performances of the algorithms between (a) and (b) in terms of time complexities.

3. (a) (5 pts.) Consider an $n \times m$ matrix A and $m \times 1$ matrix X. Let $Y = AX$ be a linear system. Design an algorithm to compute and output all values of Y.

4. (a) (3 pts.) Design a recursive algorithm to compute the binomial coefficient $\binom{n}{m}$. (Hint: $\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1}$ where $0 \le m \le n$ and $\binom{n}{n} = \binom{n}{0} = 1$)
   (b) (5 pts.) Design an iterative (non-recursive) algorithm to compute $\binom{n}{m}$.

5. (a) (2 pts.) Give the magic square where "1" has been predefined at the center of the bottom row.

		1		

   (b) (6 pts.) Design an algorithm that reports (a) as the solution to the magic square.
   (c) (2 pts) What is the summation value of each row/column/diagonal of an 13×13 magic square?

6. Consider two row-major arrays A and B. Let the starting address be α and β, respectively, and the size of each element in A or B be l.
   (a) (2 pts) Find the address of $A[i][j]$ in terms of α and l, where A is a $u_1 \times u_2$ 2-D array.
   (b) (3 pts) Find the address of $B[i_1][i_2] \ldots [i_n]$ in terms of β and l, where B is a $u_1 \times u_2 \times \ldots \times u_n$ n-D array.
   (c) (3 pts) How many multiplications ($\times$) and additions ($+$) are needed to compute the address in (b).

7. Let `SWAP(a, b, c)` be a macro performing the swapping of values between variables a and b. Function `p` is defined as follows.

```
void p(char *list, int k, int n)
{  int i, tmp;
   if (k == n-1) cout << list << endl; //output list
   else
   { for (i=k; i<n; i++)
     { SWAP(list[k], list[i], tmp); // list[0]: the first element
       cout << “=>” << i << “, ” << k << “, ” << list << endl;
       p(list, k+1, n);
       SWAP(list[k], list[i], tmp);
       cout << “<=” << i << “, ” << k << “, ” << list << endl;
     }
   }
}
```

Let `list = "ABCD";` be set in the main program. What would be printed out
(a) (1%) after function `p(list, 3, 4);` is called?
(b) (5%) after function `p(list, 1, 4);` is called?

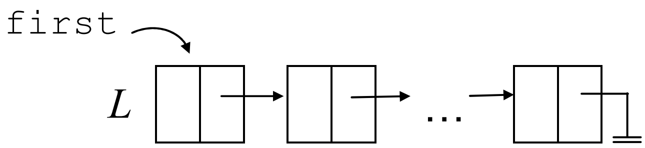
8. Consider a 0/1 matrix *M* as a maze where 0's entries represent open paths and 1's the barriers. You are developing a program to train a rat passing through the maze from the entrance (*sx*, *sy*) to the exit (*ex*, *ey*) in *M*.
(a) (2%) Specify the necessary data structures and explain their purposes.
(b) (7%) Design an algorithm for this rat.
(b) (1%) What is the time complexity of your algorithm?

9. Let `A^B` denote A^B in which the priority of ^ is higher than that of `* and /`.
(a) `A+(B-C)*D^(E+F/G)+H`
(2 pts.) Give the postfix notation of (a).
(b) `QRSTUVW-^*X/+*+`
(2 pts.) Give the prefix notation of (b).

Suppose that functions “push” and “pop” for strings have already been defined. You may call them directly. Please design algorithms for
(c) (5 pts.) transforming the infix notation into the postfix notation;
(d) (3 pts.) transforming the postfix notation into the prefix notation;
(d) (3 pts.) transforming the prefix notation into the postfix notation;

10. Based on the following declaration, a singly linked list *L* would be constructed.

```
struct node
{ int data;
  struct node * next;
};
struct node  * first;
```



Given a pointer *x* pointing to a certain node in *L*,

(a) (4 pts.) design a function to insert a new node storing an input integer named `element` right after the node pointed by *x*. (Hint: *x* may be a null pointer `NULL`.)

```
struct node * insertAfter(struct node * x, int element)
```

(d) (4 pts.) design a function to delete the node after the one pointed by *x*. (Hint: *x* may be `first`, or a null pointer `NULL`)

11. (10 pts.) Consider the following program segments.

[A]
```
int data[10][10], i, j;
for (i = 0; i < 10; i++)
    for (j = 0; j < 10; j++)
        data[i][j] = 1000-10*i-j-1;
```

Suppose that we obtain `8808000` when `(int) &data[0][0]` is printed and we have `999` when `data[0][0]` is printed. What values would be printed when the following variables are printed as integers?

(a) `data[1]`
(b) `(int)data[1]+1`
(c) `data[1][0]`
(d) `&data[1][1]`
(e) `*(data[0]+1)`

[B]
```
int ** data, i, j;
 data = new int * [10];
 for (i=0; i<10; i++) data[i] = new int [n];
 for (int i=0; i<m; i++)
     for (j = 0; j < n; j++)
         data[i][j] = 100+10*i+j;
```

Suppose that we obtain `3906000` when `(int) &data[0][0]` is printed,
`3906048` `(int) &data[0][0]`
`3906096` `(int) &data[1][0]`

`100` `data[0][0]`

What the values would be when the following variables are printed as integers?

(f) `data[1]+1`

(g) `data[1][0]`

(h) `*(data[0]+1)`

(i) `&data[1][1]`

(j) `*(*(data+1)+1)`

12. A string S consisting of four characters: '(', ')', '[' and ']' is *parenthese-balanced* if any of the following rules is met:

(1) S is an empty string.

(2) If α is parenthese-balanced, then (α) and $[\alpha]$ are parenthese-balanced.

(3) If α and β are parenthese-balanced, then $\alpha\beta$ is parenthese-balanced.

(a) (4 pts.) Design an algorithm to determine whether input string S is parenthese-balanced.

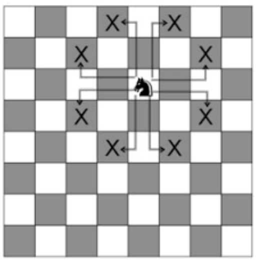
Input: S consisting of members in { '(', ')', '[', ']' }

Output: 1 if S is parenthese-balanced; 0 otherwise

(b) (1 pts.) What is the time complexity of your algorithm?

13. In a classic chessboard, the possible moves of a knight is as Fig. 1(a). Given an $n\times n$ board B, a *knight's tour*, starting from any square, traverses all squares on B by knight's moves. Let each square be indexed as j when the knight takes the jth move here. Specifically, a tour 1-2-3- ... -n^2 on B is a knight's tour if i-(i+1) is a legal knight's move for $1\le i\le n^2-1$. Fig.1(b) shows 3 knight's tours in a 5×5 board and Fig. 1(c) gives a knight's tours on a 8×8 board.

(a)



(b)

1	22	13	18	7
14	19	8	23	12
9	2	21	6	17
20	15	4	11	24
3	10	25	16	5

1	14	9	20	3
24	19	2	15	10
13	8	25	4	21
18	23	6	11	16
7	12	17	22	5

1	14	9	20	3
24	19	2	15	10
13	8	23	4	21
18	25	6	11	16
7	12	17	22	5

(c)

0	59	38	33	30	17	8	63
37	34	31	60	9	62	29	16
58	1	36	39	32	27	18	7
35	48	41	26	61	10	15	28
42	57	2	49	40	23	6	19
47	50	45	54	25	20	11	14
56	43	52	3	22	13	24	5
51	46	55	44	53	4	21	12

Fig.1 Knight's moves and knight's tours

(a) (8 pts.) Design an algorithm to determine whether the n^2 indices on the n^2 squares of a given $n \times n$ board B constitute a knight's tour.

(b) (2 pts.) What is the time complexity of your algorithm?