※ 考生請注意：本試題 □可 ☑不可 使用計算機

I. Multiple choice and Short answer questions

(1) Suppose we are sorting an array of eight integers using quicksort, and we have just finished the first partitioning with the array looking like this: (5%)

2 5 1 7 9 12 11 10

Which statement is correct? 5%

A. The pivot could be either the 7 or the 9.
B. The pivot could be the 7, but it is not the 9.
C. The pivot is not the 7, but it could be the 9.
D. Neither the 7 nor the 9 is the pivot.

(2) How many linked lists are used to represent a graph with n nodes and m edges, when using an edge list representation?(5%)

A. m
B. n
C. m + n
D. m*n

(3) Assume that a queue is implemented with a linked list, keeping track of a front node and a rear node with two reference variables. Which of these reference variables will change during an insertion into a NONEMPTY queue?(5%)

A. Neither changes
B. Only front changes.
C. Only rear changes.
D. Both change.

(4) Given an infix expression: 4+3*(6*3-12). Suppose that we are using the usual Stack algorithm to convert the expression from infix to postfix notation. What is the maximum number of symbols that will appear on the stack AT ONE TIME during the conversion of this expression?(5%)

A. 1
B. 2
C. 3
D. 4
E. 5

(5) In hashing, what is the bucket size, which will lead to the situation that the overlapping and collision occur at the same time? Why is the answer?(5%)

(6) What is the minimum number of queues needed to implement a heap? Why is the answer?(5%)

(7) There are 15, 9, 31, 16, and 32 nodes in 5 different trees. Which one of them can form a full binary tree? Why is the answer?(5%)

(8) Given the sequence 4 15 8 18 9 11 10. You are asked to reorder it so that when elements are inserted in this new order into an empty binary search tree, the resulting tree has height 3. Note that the height of a binary search tree indicating the number of nodes in the longest path from root to leaf.(5%)

(背面仍有題目,請繼續作答)

※ 考生請注意：本試題 □可 ☑不可 使用計算機

(9) Given the following recursive function, `recursive_f`. Assume that the running time of the function call `linear(n)` is O(n). What is the running time of `recursive_f` as a function of n? Why is the answer? (10%)

```
void recursive_f (int n)
{
    if (n == 0)
        println("End");
    else
    {
        recursive_f (n/2);
        recursive_f (n/2);
        linear(n);
    }
}
```

II. This part contains two problem sets:

(10) (28%) True or False, and EXPLAIN
Answer T or F for each of the following statements to indicate whether the statement is true or false, respectively. If the statement is correct, briefly state why. If the statement is wrong, explain why or give a counter example. Answers without reasons will get at most 1 point.

(a) (T , F) Given an undirected simple graph $G = (V, E)$, if $|E| \geq |V| - 1$, then G is connected.
(b) (T , F) Given an undirected simple connected graph $G = (V, E)$ where each edge $(i, j) \in E$ has a unique weight c_{ij}, then the minimum spanning tree has to be unique.
(c) (T , F) Let A be an sorted array of n distinct real numbers, constructing a balanced binary search tree for A requires $O(n)$ time.
(d) (T , F) The Dijkstra's algorithm can be applied on any digraph $G = (V, E)$ that contains no negative cycle but some edge (i, j) has negative length $c_{ij} < 0$.
(e) (T , F) You must shift items in an array implementation of a queue in either the enqueue or dequeue operation.
(f) (T , F) Solution to the recurrence $T(n) = 8T(\frac{n}{2}) + \sqrt{n}$ is $\Theta(n^3)$
(g) (T , F) If you want to keep track of appointments and other events organized by their time and date. It is important to be able to efficiently find all the events between a starting time/day and an ending time/day. To find these events efficiently, a binary min-heap is better than a binary search tree to store these appointments or events.

(11) (22%) A tournament $T = (V, E)$ is a simple digraph of $|V| = n$ vertices and $|E| = \frac{n(n-1)}{2}$ edges , suppose you already know *outdeg*[i], the outdegree for each vertex i. A tournament is *transitive*, whenever edge $(u, v) \in E$ and $(v, w) \in E$ implies $(u, w) \in E$. In other words, if there exists any 3 vertices i, j, k in T with edges (i, j), (j, k), and (k, i), then T is NOT transitive. Now you want to check whether T is transitive or not.
(a) [6%] Give an $O(n^3)$ time method to do so. You should briefly explain steps of your method and its complexity.
(b) [8%] Give an $O(n^2)$ or $O(n \log n)$ time method to do so. You should briefly explain steps of your method and its complexity.
(c) [8%] Can this be done in $O(n)$ time? If not, explain why it is not possible. Otherwise, briefly explain steps of your method and its complexity.