[作答注意事項：作答時請將各題之完整題號(例：*(1-b)*或*(2-a)*等等)標示清楚。]

(1) Suppose a list L of component objects A, B and C is expressed as $L = (A\ B\ C)$. An empty list expressed as ( ) has no component object. A component object of a list can be either a symbol or a list, for example, in $L' = (A\ (B\ C)\ D)$ the component object $(B\ C)$ is a list. The depth of list L, denoted as Depth(L), is defined as:

$$\text{Depth}(L) = \begin{cases} 0, & \text{if } L \text{ is a non-list symbol} \\ 1, & \text{if } L \text{ is an empty list} \\ (\text{max depth among all component objects of } L) + 1, & \text{otherwise.} \end{cases}$$

For example, if $L' = (A\ (B\ C)\ D)$ then Depth(L') = 2 if none of A, B, C, D is a list.

Based on the above, please do the following: (40%, total 40 points)

(1-a) Design a data structure to implement the list structure described above. (5 points)

(1-b) Based on your design in *(1-a)*, develop an algorithm to determine whether two given lists $L1$ and $L2$ are equal. (10 points)

(1-c) Based on your design in *(1-a)* , develop an algorithm to determine the depth of a given list $L3$. (10 points)

(1-d) What is the time complexity of your algorithm developed in *(1-c)*? (5 points)

(1-e) Based on your design in *(1-a)*, develop an algorithm to flat a given list, for example, if $L4 = (A\ (B\ (C\ (D\ E)))\ (F)\ G)$, then your algorithm should flat $L4$ and produces a list $(A\ B\ C\ D\ E\ F\ G)$. (10 points)

[Note: Any popular programming language or pseudo code can be used in your answers.]

(2) Explain the following terms: (30%, 3 points each)

(2-a)	Balanced tree	*(2-b)*	ADT
(2-c)	Loading factor	*(2-d)*	Skip list
(2-e)	Horner's rule	*(2-f)*	Topological sorting
(2-g)	Spanning tree	*(2-h)*	Connected component
(2-i)	$\Omega(n^2)$	*(2-j)*	m-way search tree

(3) Answer the following: (10%, total 10 points)

(3-a) What is the average time complexity of the quick sort algorithm? Give an explanation briefly. (5 points)

(3-b) What is the worst-case time complexity of the quick sort algorithm? (2 points)

(3-c) Under what conditions will the worst-case of the quick sort algorithm happen? Give an explanation briefly. (3 points)

(背面仍有題目,請繼續作答)

[作答注意事項：作答時請將各題之完整題號(例：*(4-b)*或*(4-c)*等等)標示清楚。]

(4) Multiple choices: (20%, 4 points each)
[For the following questions, you must give exact all proper items to get the points.]

(4-a) Which is(are) true for heap sort?

(A) an unstable sorting algorithm (B) comparison-based sorting algorithm
(C) time complexity $O(\log n)$ (D) time complexity $\Omega(n)$
(E) space complexity $O(n \log n)$ (F) None of the above.

(4-b) What is(are) true for hashing?

(A) Hashing can not be used as one sorting algorithm.
(B) Hashing always gives constant searching time.
(C) A hash table may be implemented using only array structure.
(D) The number of buckets in a hash table must be a prime number.
(E) It is possible to avoid collision and overflow without any overheads.
(F) None of the above.

(4-c) Given a connected undirected graph $G = (V, E)$ and $|V| > 1$, let *BFS(i)* and *DFS(i)* denote the outcomes of visiting all nodes in G starting from node i by *breadth-first search* and *depth-first search* respectively. Which is(are) true?

(A) It is possible that $BFS(j) = DFS(k)$. (B) It is possible that $BFS(i) = DFS(i)$
(C) It is impossible that $BFS(j) = BFS(k)$. (D) A spanning tree of G has $|V|$ edges.
(E) G has exact $(2^{|V|-1} - 1)$ spanning trees. (F) None of the above.

(4-d) For a non-empty complete binary tree T, let n denote the total number of nodes in T. The number of leaf nodes in T is denoted as L. Let n_1 and n_2 denote the number of degree-1 nodes and degree-2 nodes respectively, and h denotes the height of T. Which is(are) true?

(A) $h \le L + n_1$ (B) $\lfloor n_1 / 2 \rfloor = 0$
(C) $(n_1 + n_2) \le L \le (n_1 + n_2 + 1)$ (D) $n_1 + 2n_2 = n - 1$
(E) $n_1 < L \le 2^{(h-1)}$ (F) None of the above.

(4-e) Which among the following is(are) true?

(A) Using threaded binary search tree can speed up the searching process.
(B) If $k \geqq 1$ then $\log^k n = O(n)$.
(C) Given the pre-order sequence and the leaf nodes of a binary tree, the binary tree can be uniquely defined.
(D) The height of an AVL tree of N nodes is $\Theta(\log_2 N)$.
(E) When implementing an undirected graph G, the storage space required by an adjacency matrix is always less than that of an adjacency list.
(F) The spanning tree of a graph is a bipartite graph.