考試科目	資料結構及演算法	所別	8141 資訊科學系	考試時間	3 月 6 日(六) 第 一 節

說明：1. 請書寫必要之解題過程。過程正確但答案錯誤，可能有部分分數。如題目之解答非顯而易見者，僅書寫答案而缺乏必要之過程，亦無法獲得該題之滿分。

2. 可使用中文或英文作答。

1. (16%, 2% for each problem) 是非題(True of False): (本大題僅回答 T 或 F 即可，不需理由)

子題號	題目內容
(1)	Hashing is a technique to achieve an O(1) expected search time. However, its worst-case search time is O(log n).
(2)	Adjacency matrix is good for representing a sparse graph.
(3)	$2^{2n} = O(2^n)$, where O is the notation for asymptotic upper bound.
(4)	Membership test in a linked list requires O(log n) time, in the worst case, for input size n.
(5)	We need two pointers to implement either queue or stack using linked lists.
(6)	The best case time complexity of a selection sort algorithm is the same as that of a insertion sort algorithm.
(7)	The worst case of insertion operation on a binary search tree takes O(log n).
(8)	A binary tree can not be used to represent a parent with three children.

2. (24%, 3% for each problem) 選擇題(select the correct answer): (本大題僅選擇答案即可，不需理由)

子題號	題目內容
(1)	What is the complexity of the recurrence equation $T(n) = 5\,T(\frac{n}{2}) + \Theta(n^2)$? (A) $\Theta(n^{\log_2 5})$ (B) $\Theta(n^2)$ (C) $\Theta(n^{\log_5 2})$ (D) $\Theta(n^2 \log n)$
(2)	If the complexity of the recurrence equation $T(n) = T(\frac{n}{3}) + T(\frac{2n}{3}) + f(n)$ has been determined as $\Theta(n^2 \log_2 n)$, what will be the complexity of $T(n) = T(\frac{n}{10}) + T(\frac{9n}{10}) + f(n)$? (A) $\Theta(n^2 \log_2 n)$ (B) $f(n)$ (C) $\Theta(n^2 \log_9 n)$ (D) Can't be determined.

備註	試題隨卷繳交

考試科目	資料結構及演算法	所別	資訊科學系	考試時間	3 月 6 日(六) 第 一 節

2. (接續前頁題 2)

子題號	題目內容
(3)	How many edges will be a minimum spanning tree of a n-node graph? (A) n (B) n^2 (C) O(log n) (D) n - 1
(4)	The time complexity of the inorder-tree- traversal algorithm is (A) O(n) (B) O(n log n) (C) O(log n) (D) $O(n^2)$
(5)	Which of the following algorithm has the stack property? (We generally call the last in first out property as the stack property.) (A) breadth-first search (B) depth-first search (C) preorder-tree-traversal (D) 以上皆非
(6)	There are two most popular algorithms in finding the minimum spanning trees, the Kruskal's algorithm and the Prim's algorithm. Which of the following statement about these algorithms is correct? (A) They both are greedy algorithms. (B) During the MST finding process, the intermedium solutions of both algorithms are always a tree. (C) The time complexity of both algorithms, in the worst case, is O(E log E), where E is the number of edges in the given graph. (D) The time complexity of both algorithms, in the worst case, is $O(E^2 \log E)$, where E is the number of edges in the given graph.
(7)	In the following list of problems, not all of them have known polynomial time solutions: (1) the shortest path problems, (2) the longest path problems, (3) 2-CNF satisfiability, (4) 3-CNF satisfiability, (5) Euler tour (traverses each edge of a directed graph exactly once), (6) Hamiltonian cycle (traverses each vertex of a directed graph exactly once). Which of the following statement is correct? (A) Problems (1), (3), and (5) are NPC problems. (B) Problems (1), (3), and (5) can be solved in polynomial time. (C) Problems (1), (3), and (6) can be solved in polynomial time. (D) They all are NPC problems.

備註	試題隨卷繳交

考試科目	資料結構及演算法	所別	資訊科學系 8141	考試時間	3 月 6 日(六) 第 一 節

2. (接續前頁題 2)

子題號	題目內容
(8)	Which of the following statements is correct? (A) One of the most important reasons to use hashing technique is to obtain O(log n) search time. (B) One of the most important reasons to use tree data structure is to obtain O(log n) tree manipulation time. (C) Binary search trees are always balanced. (D) Heap can only be implemented by linked lists.

3. (10%) NP-completeness

(a) (4%) Give the definition of the class of the "NP-complete" problems.
(b) (6%) Prove the statement: "If a NP-complete problem has a polynomial time solution, then all the NP-complete problems can be solved in polynomial time."

4. (15%) Quicksort and recurrence

For the following problems, assume the input size is n.
(a) (3%) Formulate the recurrence equation for Quicksort. (Explain every parameter you used.)
(b) (6%) Using the recurrence equation, find the best case time complexity of Quicksort.
(c) (6%) Using the recurrence equation, find the worst case time complexity of Quicksort.

5. (15%) Sorting lower bound

(a) (8%) Prove the lower bound of the comparison based sorting algorithms.
(b) (2%) The radix sort algorithm can be summarized as follows:

RADIX_SORT(A, m)
1 **for** $i \leftarrow 1$ **to** m
2 **do** use a stable sort to sort array A on digit i

What is the time complexity of this algorithm? (Assume that the maximal number of digits of all the inputs is m, each digit ranges from 0 to k, and the total number of inputs is n.)
(c) (5%) Is there any conflict on the results in (a) and (b)? Justify your answer.

備註	試題隨卷繳交

考試科目	資料結構及演算法	所別	資訊科學系 8141	考試時間	3 月 6 日(六) 第 一 節

6. (8%) Order of magnitude

One can generally use the big-O, big-Ω, little-o and the little-ω notations for the asymptotic upper bounds or lower bounds. The definitions of these notations are:

$$O(g(n)) = \{f(n) \mid \exists c, n_0 \text{ s.t. } 0 \le f(n) \le cg(n) \ \forall n \ge n_0\}$$
$$\Omega(g(n)) = \{f(n) \mid \exists c, n_0 \text{ s.t. } 0 \le cg(n) \le f(n) \ \forall n \ge n_0\}$$
$$o(g(n)) = \{f(n) \mid \forall c, \exists n_0 \forall n \ge n_0, 0 \le f(n) \le cg(n)\}$$
$$\omega(g(n)) = \{f(n) \mid \forall c, \exists n_0 \forall n \ge n_0, 0 \le cg(n) \le f(n)\}$$

(a) (4%) Give two functions that is in $O(n^2)$ but not in $o(n^2)$.

(b) (4%) Give two functions that is in $\Omega(n^2)$ but not in $\omega(n^2)$.

7. (12%) Graph Algorithms

The Ford-Fulkerson algorithm can be used to find the Maximum Flow (MF) of a given graph.

The simplest version of the Ford-Fulkerson algorithm is:

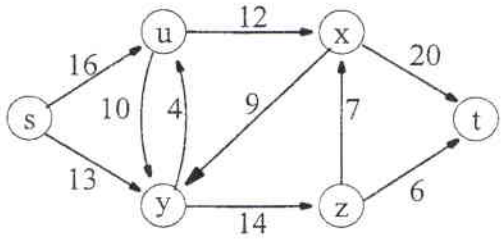
initialize flow f to 0
while there exists an augmenting path p
  do augment flow f along p
return f

and can be refined as:

FORD-FULKERSON(G,s,t)
for each edge $(u,v) \in E[G]$
  do $f[u,v] \leftarrow 0$ and $f[v,u] \leftarrow 0$
while there exists a path p from s to t in the residual network G_f
  do $c_f(p) \leftarrow \min\{c_f(u,v) : (u,v) \text{ is in } p\}$
    for each edge (u,v) is in p
      do $f[u,v] \leftarrow f[u,v] + c_f(p)$

(a) (4%) What is the complexity of this algorithm in terms of the graph size V and E.

(b) (4%) Using this algorithm to find the maximum flow of the following graph.



(c) (4%) Justify your answer in (b)