國立臺北科技大學九十九學年度碩士班招生考試

系所組別：2310 資訊工程系碩士班甲組

第二節　離散數學與演算法　試題

注意事項：
1. 本試題共十題，配分共 100 分。
2. 請標明大題、子題編號作答，不必抄題。
3. 全部答案均須在答案卷之答案欄內作答，否則不予計分。

1. (10 pts) For the following problems, please select the correct answers. In each problem, you must have all the correct ones selected to have the full credits. Otherwise, no credits will be given.
   (1) (5 pts) Which of the following linear equations CANNNOT be solved in integers?
   (a) $154x + 260y = 5$
   (b) $108x + 30y = 7$
   (c) $45x + 14y = 1$
   (d) $621x + 736y = 46$

   (2) (5 pts) Which of the following statements is true?.
   (a) The intersection of two equivalence relations is again an equivalence relation.
   (b) The union of two equivalence relations is again an equivalence relation.
   (c) If R_1 and R_2 are two symmetric relations on a set, then so is $R_1 \circ R_2$.
   (d) If R is a reflexive and transitive relation, then $R \circ R$ is transitive.
   Note that, giving two relations R and S where R is from set A to set B and S is from set B to set C. The composition of R and S, denoted by $R \circ S$, is the relation from A to C and, for all $a \in A$, $c \in C$, $a(S \circ R)c$ if there exists some $b \in B$ such that aRb and bSc.

2. (10 pts) Consider the following expressions in an infix form:
   $$6 + 5 * 3 - 4 - 8 + 6 * 2$$
   (1) (3 pts) Please convert it in a postfix form.
   (2) (3 pts) Please convert it in a prefix form.
   (3) (4 pts) Please use the postfix form to evaluate this expression. You NEED to draw the sequence of stack configurations in the evaluation.

3. (10 pts) For a binary tree T, we use $|T|$ to denote the number of nodes of T. If $|T| = 0$ or $|T| = 1$, there is only one binary tree. If $|T| = 2$, then there are two distinct binary trees. Please answer the following questions:
   (1) (2 pts) When $|T| = 3$, how many distinct binary trees are there? Please draw them.
   (2) (5 pts) In general, we would like to find the total number of distinct binary trees when $|T| = n$. Let B_n denote this number. Please write a recurrence equation of B_n.
   (3) (3 pts) Please obtain B_n in explicit form using the recurrence equation in (2).

4. (10 pts) Show that the language
   $$L = \{ a^m b^m c^m \mid m \geq 1 \}$$
   over the alphabet $\Sigma = \{a, b, c\}$ is not a context-free language.

5. (10 pts) Solve the recurrence relation
   $a_n - 6a_{n-1} + 9a_{n-2} = 2^n$, if $n \geq 2$, with initial conditions, $a_0 = 4$ and $a_1 = 9$.

6. (10 pts) Use the master method to give tight asymptotic bounds for the following recurrences. Please justify your answer.
   (1) (2 pts) $T(n) = T(n-1) + \frac{1}{n}$
   (2) (2 pts) $T(n) = 5T(\frac{n}{2}) + n^3$
   (3) (2 pts) $T(n) = 4T(\frac{n}{2}) + n$
   (4) (2 pts) $T(n) = 3T(\frac{n}{2}) + n^2$
   (5) (2 pts) $T(n) = 4T(\frac{n}{2}) + n^3$

注意：背面尚有試題

7. (10 pts) Finding the closest pair of points in a set Q of $n > 3$ points. For a divide-and-conquer algorithm for this problem, each recursive invocation of the algorithm takes as input a subset $P \subseteq Q$ and arrays X and Y, each of which contains all the points of the input subset P. The points in array X are sorted so that their x-coordinates are monotonically increasing. Similarly, array Y is sorted by monotonically increasing y-coordinate. Use "presorting" to maintain this sorted property without actually sorting in each recursive call. The recursive invocation with inputs P, X, and Y carries out the divide-and-conquer paradigm as follows.

Divide:
It finds a vertical line l that bisects the point set P into two sets P_L and P_R such that $|P_L| = |P|/2$, $|P_R| = |P|/2$, all points in P_L are on or to the left of line l, and all points in P_R are on or to the right of line l. The array X is divided into arrays X_L and X_R, which contain the points of P_L and P_R respectively, sorted by monotonically increasing x-coordinate. Similarly, the array Y is divided into arrays Y_L and Y_R, which contain the points of P_L and P_R respectively, sorted by monotonically increasing y-coordinate.

Conquer:
Having divided P into P_L and P_R, it makes two recursive calls, one to find the closest pair of points in P_L and the other to find the closest pair of points in P_R. The inputs to the first call are the subset P_L and arrays X_L and Y_L; the second call receives the inputs P_R, X_R, and Y_R. Let the closest-pair distances returned for P_L and P_R be δ_L and δ_R, respectively, and let $\delta = \min(\delta_L, \delta_R)$.

Combine:
The closest pair is either the pair with distance δ found by one of the recursive calls, or it is a pair of points with one point in P_L and the other in P_R. The algorithm determines if there is such a pair whose distance is less than δ. Observe that if there is a pair of points with distance less than δ, both points of the pair must be within δ units of line l. Thus, they both must reside in the 2δ-wide vertical strip centered at line l.

(1) (5 pts) Find the running time of each recursive step. Please justify your answer.
(2) (5 pts) Find the running time of the entire algorithm. Please justify your answer.

8. (10 pts) Consider sorting n numbers stored in array A by first finding the smallest element of A and exchanging it with the element in A[1]. Then find the second smallest element of A, and exchange it with A[2]. Continue in this manner for the first n-1 elements of A.
(1) (4 pts) Write pseudocode for this algorithm, which is known as selection sort.
(2) (2 pts) What loop invariant does this algorithm maintain?
(3) (2 pts) Why does it need to run for only the first n-1 elements, rather than for all n elements?
(4) (2 pts) Give the best-case and worst-case running times of selection sort in Θ-notation.

9. (10 pts)
(1) (5 pts) Define NP-complete languages, which are the hardest problems in NP.
(2) (5 pts) Proof that the circuit-satisfiability problem belongs to the class NP.

10. (10 pts) Given a graph $G = (V, E)$ and a distinguished source vertex s, breadth-first search systematically explores the edges of G to "discover" every vertex that is reachable from s. It computes the distance (smallest number of edges) from s to each reachable vertex. It also produces a "breadth-first tree" with root s that contains all reachable vertices.

To keep track of progress, breadth-first search colors each vertex white, gray, or black. All vertices start out white and may later become gray and then black. A vertex is discovered the first time it is encountered during the search, at which time it becomes nonwhite. Gray and black vertices, therefore, have been discovered, but breadth-first search distinguishes between them to ensure that the search proceeds in a breadth-first manner. If $(u, v) \in E$ and vertex u is black, then vertex v is either gray or black; that is, all vertices adjacent to black vertices have been discovered. Gray vertices may have some adjacent white vertices; they represent the frontier between discovered and undiscovered vertices.
(1) (6 pts) Write pseudocode for the breadth-first-search algorithm, BFS(G, s). Assume that the input graph $G = (V, E)$ is represented using adjacency lists. It maintains several additional data structures with each vertex in the graph. The color of each vertex $u \in V$ is stored in the variable *color*[u], and the predecessor of u is stored in the variable $\pi[u]$. If u has no predecessor (for example, if $u = s$ or u has not been discovered), then$\pi[u]$ = NIL. The distance from the source s to vertex u computed by the algorithm is stored in $d[u]$. The algorithm also uses a first-in, first-out queue Q to manage the set of gray vertices.
(2) (2 pts) What is the total running time of the BFS algorithm? Please justify your answer.
(3) (2 pts) What is the running time of BFS if its input graph is represented by an adjacency matrix and the algorithm is modified to handle this form of input?