國立臺灣科技大學
八十九學年度碩士班招生考試試題

系所組別： 電子工程系甲組
科　目： 資料結構與演算法

(總分共 100 分)

Note: When you are asked in the following questions to write an algorithm, you may use the C language or any pseudo code. *N* is a nature number.

1. (15%) Given a linked list of *N* nodes of data, write a recursive algorithm that prints all the node data in the list in the REVERSE order. Explain why its recurrence relation is: $C_N = C_{N-1} + 1$ for $N > 1$, with $C_1 = 1$, where C_N stands for the time complexity of the algorithm in printing *N* nodes. Solve C_N.

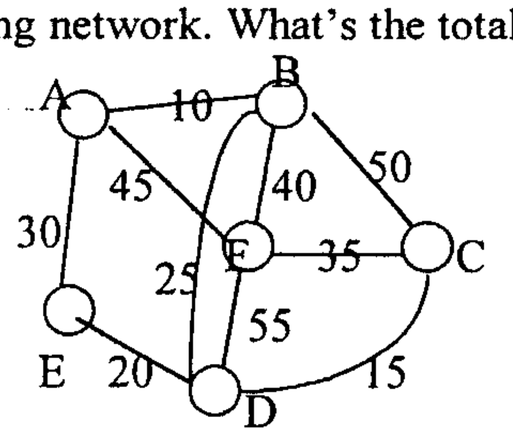
2. (15%) Given an array of *N* items of data, write a recursive Bubble Sort algorithm that sorts the array into the ascending order. Explain why its recurrence relation is: $C_N = C_{N-1} + N - 1$ for $N > 1$, with $C_1 = 0$, where C_N stands for the time complexity of bubble sorting *N* items. Prove by mathematical induction that $C_N = (N^2 - N)/2$.

3. (15%) Given an array of sorted *N* items of data, write a Binary Search algorithm that searches the array for the data that contains some given key. Prove that its worst case time complexity is about $log_2 N$.

4. (15%) Given a complete *m*-ary tree of *h*-level height, with the root designated as level *0*, write an iterative Depth-First Search algorithm that searches the tree for some given key. Prove that its worst case time complexity is $O(m^h)$.

5. (20%) Write a Divide-and-Conquer algorithm whose time complexity in processing a problem with input size *N* can be expressed by the following recurrence relation: $C_N = 2C_{N/2} + N$ for $N > 1$, with $C_1 = 0$. Solve C_N.

6. (10%) Show how the greedy Kruskal's algorithm produces a minimum spanning tree from the following network. What's the total cost of the spanning tree?



Legends:

A~F for node names, and numbers on the links for cost.

7. (10%) Show how a recursive algorithm uses Dynamic Programming to calculate F_{10}, the 10^{th} term of the Fibonacci series. Calculate and compare its time complexity on this problem instance with that of a recursive algorithm without using Dynamic Programming. Note that $F_0 = 0$, $F_1 = 1$, and $F_N = F_{N-1} + F_{N-2}$ for $N > 1$.