國立中央大學資訊工程學系110學年度第二學期博士班資格考試題紙

科目： **演 算 法** (Algorithms) 第一頁 共一頁(page 1 of 1)

1. Consider the problem of neatly printing a paragraph on a printer. The input text is a sequence of n words of lengths $l_1, l_2, ..., l_n$, measured in characters. We want to print this paragraph neatly on a number of lines that hold a maximum of M characters each. Our criterion of "neatness" is as follows. If a given line contains words i through j and we leave exactly one space between words, the number of extra space characters at the end of the line is $M - j + i - \sum_{k=i}^{j} l_k$. We wish to minimize the sum, over all lines except the last, of the cubes of the numbers of extra space characters at the ends of lines. Give a dynamic-programming algorithm to print a paragraph of n words neatly on a printer. Analyze the running time and space requirements of your algorithm. (20%)

2. Consider the following even-partition problem: Given a list of n positive integers, partition the list into two sublists, each of size $n/2$ (assume that n is even), such that the difference between the sums of the integers in the two sublists is minimized.
   a) Give a dynamic programming algorithm to solve this problem. (20%)
   b) Give a decision version of this problem, and show that it is NP-Complete. You may assume that the sum of subset problem is a known NP-Complete problem. (10%)

3. Let $S=\{s_1, s_2, ..., s_n\}$ be a non-empty set of n elements. Write an algorithm to select the media of S with the linear time complexity in the worst case. (20%)

4. The 0/1 knapsack problem is described as follows. Given the capacity m of a knapsack and n objects whose weights are $w_1,\ldots,w_n$ and whose profits are $p_1,\ldots,p_n$, find the largest value of $\sum_{1\le i\le n} p_i x_i$ by assigning either 0 or 1 to $x_1,\ldots,x_n$ under the constraint $\sum_{1\le i\le n} w_i x_i \le m$, where $w_1,\ldots,w_n$ and $p_1,..,p_n$ are positive integers. Write a dynamic programming algorithm to solve the 0/1 knapsack problem with the time complexity $O(n \cdot m)$. You should analyze your algorithm to show that its time complexity is indeed $O(n \cdot m)$. (20%)

5. How can you show that a problem is an NP-Complete problem? (10%)