Homework 3 (計算方法設計，Design and Analysis of Algorithms)

註: 所有的作業皆以紙本的方式，在截止日以前繳交給助教(上課教室或台達館 737 或 738 室)，請注意不接受遲交。All homework assignments should be submitted to the TAs (at classroom or Room 737 or 738 at Delta Building) as hard copy (handwriting or paper printout) by the due date. Please note that late assignment submissions will not be accepted.

Due date: April 26, 2024

1. (30%) The maximum subarray sum problem is to find a contiguous subarray with the largest sum within a given a one-dimensional array $A = [a_1, a_2, \dots, a_n]$ of n numbers. For example, if $A = [1, -2,5, -3,4,8, -7,6]$, then the contiguous subarray with the largest sum is $[5, -3,4,8]$ and its sum is 14. Please design a divide and conquer algorithm whose time complexity is better than $O(n^2)$ to solve the maximum subarray sum problem (15%) and analyze its time complexity (10%). Note that you can utilize the master theorem to analyze the time complexity.

2. (30%) Given a sequence $S = [a_1, a_2, \dots, a_n]$ of n different numbers, a pair of two numbers (a_i, a_j) forms an *inversion* if $i < j$ and $a_i > a_j$. For example, if $S = [2,1,4,3,6,5]$, then $(2,1), (4,3)$ and $(6,5)$ are three inversions in S. The inversion counting problem is to compute the total number of inversions in S. Please design a divide and conquer algorithm whose time complexity is better than $O(n^2)$ to solve the inversion counting problem (15%) and analyze its time complexity (10%). Note that you can utilize the master theorem to analyze the time complexity.

3. (40%) Let $A = (a_{ij})$ and $B = (b_{ij})$ be two $n \times n$ matrices. Then the product matrix $C = A \cdot B$ is also a $n \times n$ matrix and its entry c_{ij} can be computed by the following formula: $c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$. It is already known that the conventional method for computing C requires $\Theta(n^3)$ time, because it needs to compute n^2 matrix entries, each of which is the sum of n values. However, Strassen has proposed a divide and conquer algorithm that can compute C in a more efficient way than the conventional method. Please introduce the Strassen's matrix multiplication algorithm (30%) and explain why its time complexity is better than that of the conventional method (10%).