# Lab 2 Programming Problem

**Problem 1:** Implement the Divide-and-Conquer algorithm to count the number of inversions in a list of integers.

**Problem 2:** Mr. Coco buys coconuts from a farmer and sells them to a store owner. Due to a range of factors, like the changing world trade rules, climate change and the varying price of cowdung, the buying and selling rates of coconuts changes everyday. These rates are available to Mr. Coco for **n** consecutive days in two lists **B[ ]** and **S[ ]**. Assume that the price of coconuts is always a positive integer. Mr. Coco has **C** rupees (again a positive integer) to buy coconuts. He cannot of course, sell coconuts before buying them, but buying and selling on the same day are allowed. Thus all coconuts bought on Day i must be sold on Day j for some j satisfying $0 \le i \le j \le n-1$.

For reasons unknown to Mr. Coco, the farmer and store owner are only willing to trade with him once. So Mr. Coco can make ***exactly one*** buying and one selling. So he has to choose the days i and j such that his profit is maximized.

An exhaustive search over all pairs (i,j) satisfying $0 \le i \le j \le n-1$ takes $\Theta(n^2)$ running time. We can do much better using a divide-and-conquer strategy. Break the set of days into two equal-sized halves. Recursively compute the optimal profits LOPT and ROPT for the left and right halves. It is also allowed that items are bought in the first half, and sold in the second half. The optimal way to do this is to find Day i in the first half on which the buying rate is minimum, and to find Day j in the second half on which the selling rate is maximum. Let LROPT denote the profit for these choices of i and j. Then, the maximum profit of Mr. Intermediario is max(LOPT, ROPT, LROPT). This algorithm has a running time given by $T(n) = 2T(n/2)+\Theta(n)$, where the $\Theta(n)$ effort is associated with locating the minimum buying rate and the maximum selling rate in the two halves. By the master theorem, $T(n) = \Theta(n\log n)$. The space requirement is $\Theta(\log n)$ since the depth of recursion is log 2n.•

a) Write a function optTrans1*(B,S,n,C)* that implements the $\Theta(n^2)$ exhaustive search algorithm.
b) Write a function optTrans2*(B,S,n,C)* to implement the above divide-and-conquer algorithm.
c) Write a function optTrans3*(B,S,n,C)* to implement a modification of the above divide-and-conquer algorithm, achieving a running time of $\Theta(n)$. Since the recursion depth would not change, the space requirement will remain $\Theta(\log n)$

Read in the value of **n**, the lists **B[]** and **S[],** and the initial capital **C** from the user.

**Sample Output:**

```
Enter the value of n: 10
Enter the buying prices : 10 15 16 9 25 6 5 18 5 21
Enter the selling prices : 24 13 8 12 9 21 7 21 6 14
Enter the capital C: 1000

optTrans1: O(n^2) time
Buying date = 6, Buying rate = 5
Selling date = 7, Selling rate = 21
Maximum profit = 3200

optTrans2: O(nlogn) time
Buying date = 6, Buying rate = 5
Selling date = 7, Selling rate = 21

optTrans3: O(n) time
Buying date = 6, Buying rate = 5
Selling date = 7, Selling rate = 21
```