Lab 02

Deadline: 07/02/2022(11:59PM)

Course Title: **Algorithm Design** Course No: **CS 222**

In this assignment you have to write a C/C++ program that reads an $M \times N$ matrix $A[][]$ as input and computes each of the following. Note that any sub-matrix of a matrix can uniquely be identified by, using a couple of $(row, column)$ index pairs referring to elements at diagonally opposite corners respectively. We would be referring to sum of elements in a matrix X with the term *weight of* X.

1. Compute an $M \times N$ matrix $T[][]$ where each $T[i][j]$ is an ordered pair $(row_{ij}, column_{ij})$ such that $row_{ij} \leq i$ and $column_{ij} = n - 1$. In other words, $T[i][j]$ identifies a unique *non-empty* sub-matrix X_{ij} of $A[][]$ whose left bottom corner element is $A[i][j]$ and top right corner element is $A[row_{ij}][column_{ij}]$. The task in hand is to compute the value row_{ij} for each $T[i][j]$ such that the sub-matrix X_{ij} is of least weight. Note that every X_{ij} has its last column perfectly aligned with the last column of $A[][]$.
2. Compute an $M \times N$ matrix $T_s[][]$ where $T_s[i][j]$ is the sum of elements in X_{ij}.
3. Compute an $M \times N$ matrix $S[][]$ where computation of $S[i][j]$ is very similar to that of $T[i][j]$ with the only difference being $column_{ij} \geq j$ (instead of $column_{ij} = n - 1$). In other words, each $S[i][j]$ determines a unique *non-empty* sub-matrix X_{ij} of least weight (as in Task 1) that need not be perfectly aligned with the last column of $A[][]$.
4. Output the least possible s such that s is the sum of elements in a sub-matrix of A.

Assuming the following matrix $A[4][5]$, $T[2][1] = (0, 4)$ and $S[2][1] = (1, 2)$ as witnessed by the red and blue sub-matrices respectively.

6	-5	9	-3	-2
-3	2	-7	-3	-1
-2	7	2	5	4
-1	-9	1	7	-2

Your program should read from an input file *input.txt* that contains the matrix A in the following format:

- First row contains 2 values M N
- There would be M rows following the first raw, where each row contains N positive integers.

Your program should generate a file *output.txt* that contains the matrix $T[][]$, $T_s[][]$, $S[][]$ (in the format as in the file *input.txt*) and s with the solution to each task being separated by a blank line (There shouldn't be any additional text other than the expected output). You may refer Lab 01 to deal with file handling.

Design $\mathcal{O}(n^2)$ algorithms for Tasks 1 & 2 and $\mathcal{O}(n^3)$ algorithms for 3 & 4. You are supposed to submit a pdf file *ROLLNO.pdf* along with the code *ROLLNO.c*. Your pdf file should contain your algorithms for tasks 1 and 3 along with respective analyses.

**********END**********