Page No
Date

→ Pre computation (2D ARRAY)

8

Q) Given 2D array of N*N integers. Given Q queries and in each query a, b, c and d. Print sum of rectangle represented by (a, b) as top left point and (c, d) as top bottom right point.

Constraints:

$1 \le N \le 10^3$

$1 \le a[i][j] \le 10^9$

$1 \le Q \le 10^5$

$1 \le a, b, c, d \le N$

```
#INCLUDE <BITS/STDC++.h>
USING NAMESPACE STD;
CONST INT N = 1e3+10;
INT AR[N][N];

INT MAIN()
{
    INT n;
    CIN >> n;
    FOR (INT i=1; i≤n; i++)
    {
        FOR (INT j=1; j≤n; j++)
        {
            CIN >> ARR[i][j];
        }
    }
```

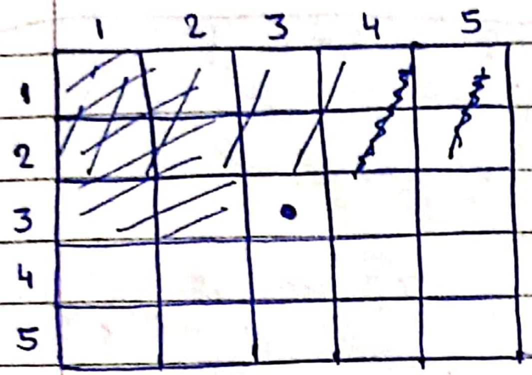
Scanned with CamScanner

```
INT Q;
CIN >> Q;
WHILE (Q--)
{
    INT A, B, C, D;
    CIN >> A >> B >> C >> D;
    LONG LONG SUM = 0;
    FOR (INT i=A; i<C; i++)
    {
        FOR (INT j=B; j<=D; j++)
        {
            SUM += AR[i][j];
        }
        COUT << SUM << ENDL;
    }

    // Time complexity -> O(N²) + O(Q*N²) ==
                10^5 * 10^6 == 10^11
    So, it will give TLE.

    RETURN 0;
}
```

// We can prevent of it from TLE by using prefix sum method. Prefix sum is storing values beforehand testing our test cases.

→ Optimized Solution:

① Approach to store values :



Let us assume we want to store value at (3, 3).
∴ The value will consist of sum of (1, 1) to (3, 3)
We can achieve this by taking sum till (2, 3) and (3, 2) and subtract till (2, 2) as it has come twice

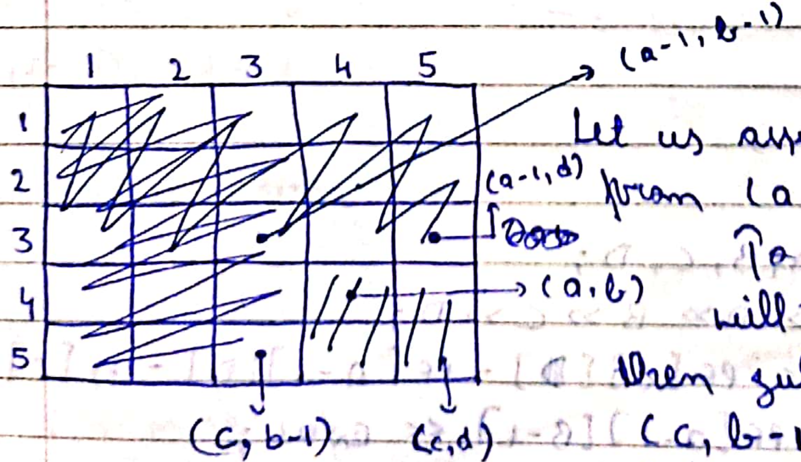
∴ $pf[3][3] = a[3][3] + pf[2][3] + pf[3][2] - pf[2][2]$

∴ General formula :

$$\boxed{pf[i][j] = a[i][j] + pf[i-1][j] + pf[i][j-1] - pf[i-1][j-1]}$$

This formula will work directly as we are taking indexing from (1, 1).

② Approach to get values :



Let us assume we want sum from (a, b) to (c, d).
To get that first we will take sum till (c, d) then sub. till (a-1, d) and (c, b-1) and add (a-1, b-1) as it has been subtracted twice.

Page No.
Date

$\therefore$ ~~PF(a,d)~~ Ans => PF[c][d] - PF[a-1][d] - PF[c][b-1] + PF[a-1][b-1]

```
#INCLUDE <BITS/STDC++.H>
USING NAMESPACE STD;
CONST INT N = 1e3+10;
INT AR[N][N];
LONG LONG PF[N][N];

INT MAIN()
{
    INT n;
    CIN >> n;
    FOR (INT i=1; i<=n; i++)
    {
        FOR (INT j=1; j<=n; j++)
        {
            CIN >> AR[i][j];
            PF[i][j] = AR[i][j] + PF[i-1][j] +
            PF[i][j-1] - PF[i-1][j-1];
        }
    }

    INT Q;
    CIN >> Q;
    WHILE (Q--)
    {
        INT A, B, C, D;
        CIN >> A >> B >> C >> D;
        COUT << PF[C][D] - PF[A-1][D] - PF[C][B-1]
            + PF[A-1][B-1] << ENDL;
```

Page No.
Date

}

// Time complexity → $O(N^2) + O(Q) == 10^6 + 10^5$
$== 10^6$

So, it won't give TLE.

RETURN 0;

}