HKOI 2016/17
JQ4 – Crosses

Alex Tung

21/1/2017

Problem description

- Rectangle with opposite vertices (0, 0) and (N, M)
- Some K points in the rectangle

- Find the number of crosses that covers all K points

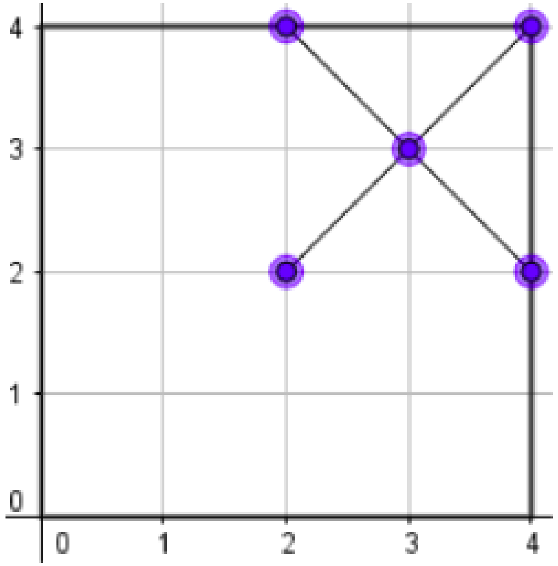
(For simplicity, assume N >= M.)

‘Crosses’

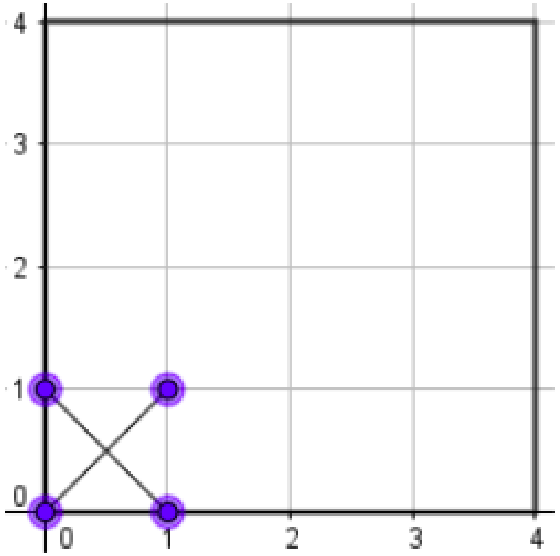
- Points on the diagonal**s** of a square

For example, if $N = M = 4$:

The set $\{(2,2),(4,2),(3,3),(2,4),(4,4)\}$ is a cross.



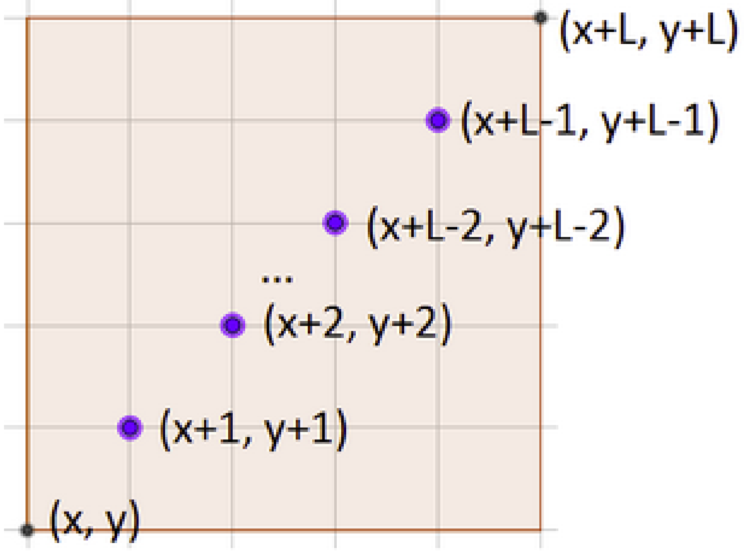
The set $\{(0,0),(0,1),(1,0),(1,1)\}$ is also a cross.



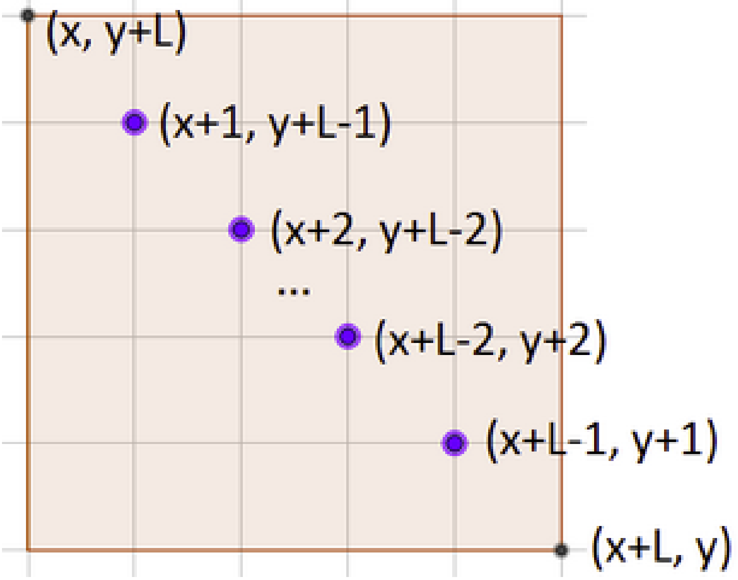
Math hints

For an axis-parallel square with length L and bottom-left corner located at (x, y):

The coordinates of the points lying on the diagonal connecting the bottom-left and the top-right corners are $(x, y), (x+1, y+1), \ldots, (x+L, y+L)$.



The coordinates of the points lying on the diagonal connecting the top-left and the bottom-right corners are $(x, y+L), (x+1, y+L-1), \ldots, (x+L, y)$.



SUBTASKS

For all cases: $1 \le N, M \le 10^6$, $0 \le K \le 200000$, $0 \le x_i \le N$, $0 \le y_i \le M$

	Points	Constraints
1	18	$1 \le N, M \le 10$
2	12	$1 \le N, M \le 120$
3	13	$1 \le N, M \le 400$
4	15	$1 \le N, M \le 3000$
5	9	$K = 0$
6	11	$K = 1$
7	22	No additional constraints

Statistics

Attempts	Max	Mean	Std Dev	Subtasks						
36	39	5.111	10.121	18: 3	12: 3	13: 0	15: 0	9: 8	11: 2	22: 0

“Probably the hardest HKOI Junior problem ever”

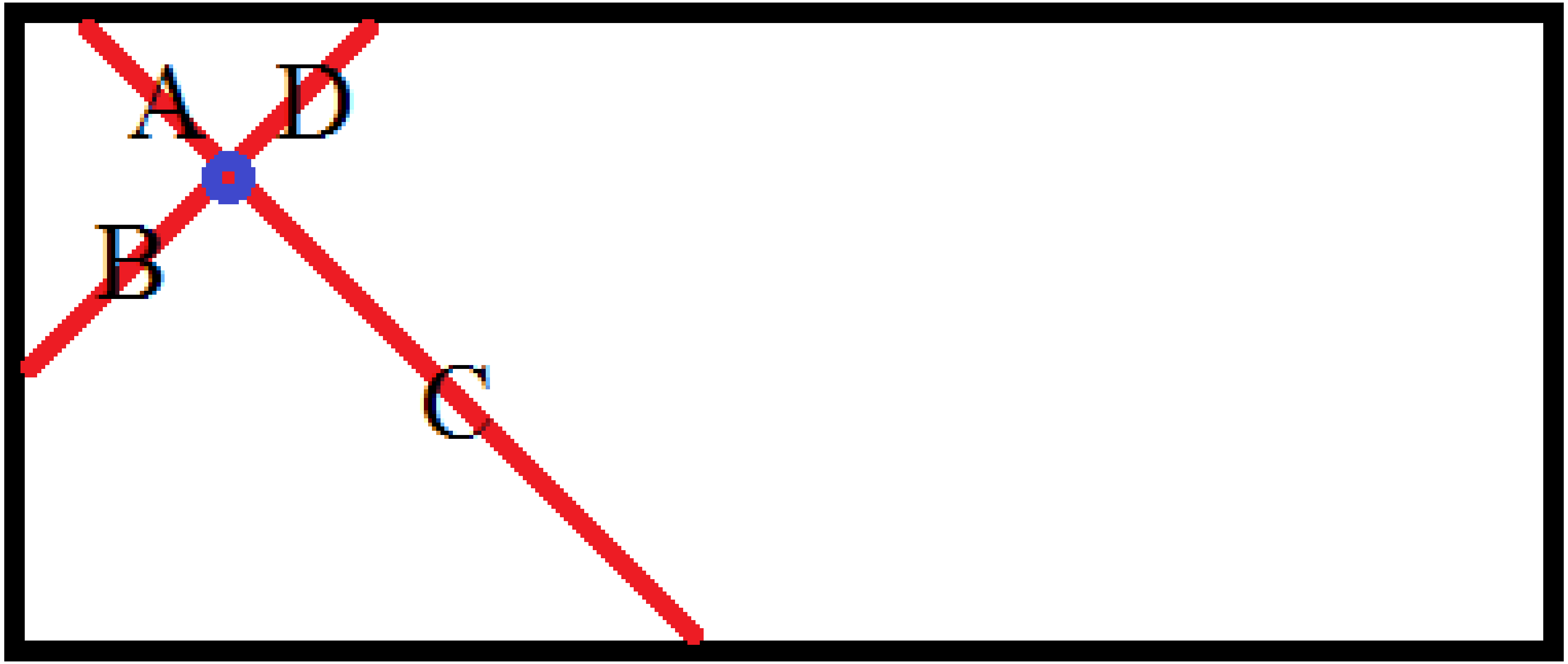
Subtask 5 (K = 0)

- Count number of squares

```
long long ans = 0;
for(int i = 1; i <= min(n, m); i++)
    ans += 1LL * (n - i + 1) * (m - i + 1);
```

- From now on, assume K > 0

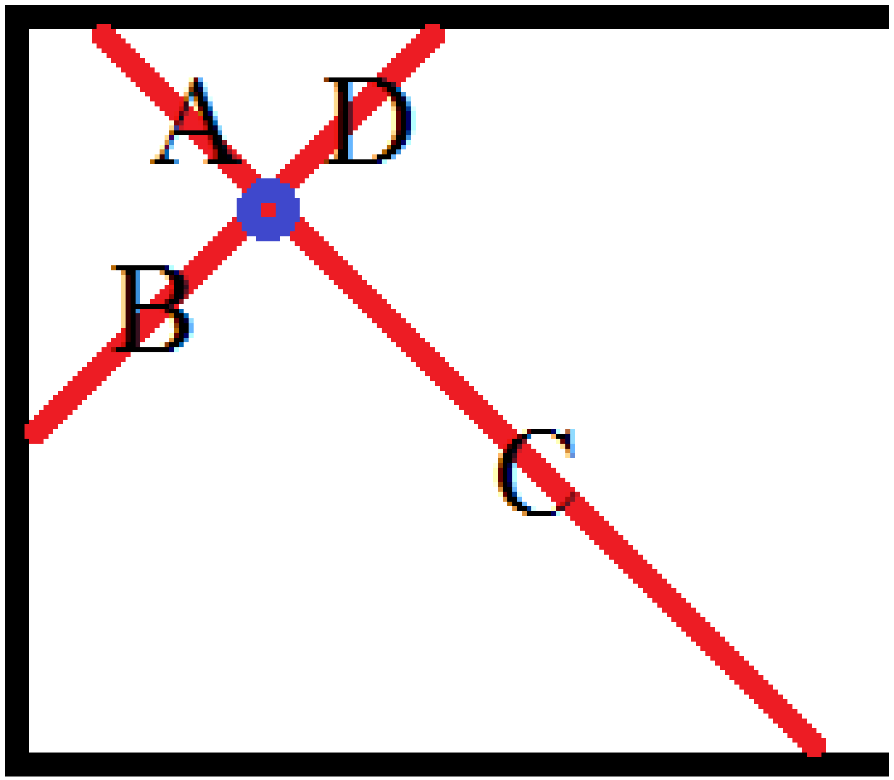
Subtask 6 (K = 1)



Formula

```
X = (A + 1) * (C + 1) - 1
Y = (B + 1) * (D + 1) - 1
Z = min(A, B, C, D)

Answer = X + Y - Z
```



Subtask 1 (N <= 10)

Algorithm:

- Use an array (S[][]) to store the K points
- For each square, mark its diagonal points on a 2D-array (say, T[][])
- Check if there is a point (x, y) s.t. S[x][y] = 1 but T[x][y] = 0
  - If no, then we have a valid cross. Add answer by 1

Time complexity

- Suppose N >= M for simplicity
- Exhaust each square: O(NM) * O(M)
- Mark points and check: O(NM)

Overall: $O(N^2M^3)$

Subtask 2 (N <= 120)

We only need to check the diagonal points!

Algorithm:

- Use an array (S[][]) to store the K points
- For each square, count how many points lie on its diagonals
  - If answer = K, we have a valid cross. Add answer by 1

Time complexity

- Exhaust each square: O(NM) * O(M)
- Count points on diagonals: O(M)

Overall: $O(NM^3)$

Subtask 3 (N <= 400)

Two ways to proceed:

1. Consider fewer squares
2. Speed up counting

Method 1: consider fewer squares

Since $K > 0$, consider a point (x_0, y_0)

Observation from the case $K = 1$:

- The **center** of the square must lie on the same diagonal (slope = +1/-1) as (x_0, y_0)!
- Only need to check $O(M^2)$ squares

Time complexity: $O(M^3 + K)$

Method 2: speed up counting

- Count points in O(1)
- Partial sum on the diagonals

CAUTION: The point on the center of a square may be double-counted

Time complexity: $O(NM^2)$

Subtask 4 (N <= 3000)

Again, two ways to proceed:

1’. Combine the two methods for subtask 3
2’. Fix center, extend square

Method 2’ leads to full solution

Method 1’: 1 + 2

- Check $O(M^2)$ squares
- $O(1)$ calculation

Time complexity: $O(M^2 + K)$

Method 2’: Extend square

- Step 1: calculate number of points on each diagonal (slope = +1/-1) and store points in S[][]

- Step 2: Exhaust positions of center

- Step 3: Check if the corresponding diagonals contain all K points
  - If not, continue;

- Step 4: If yes, find the smallest square containing the K points

Time complexity

- We consider O(NM) centers
- Note that we enter step 4 O(M) times
- Finding the smallest square takes O(M) time

Overall: O(NM)

Full solution

- Finding the smallest square in O(1) time (after some preprocessing)

For each diagonal **D**, precalculate (in O(N) time):

- Number of points on **D**
- The smallest and largest x-coordinates among the points on **D**

Full solution

- Fix a point (x_0, y_0)
- Recall: “The **center** of the square must lie on the same diagonal (slope = +1/-1) as (x_0, y_0)!”
- O(M) centers to consider

- Using the preprocessed information, calculate the smallest square containing the K points in O(1)

- Time complexity: O(N + K)

Points to note

- The center of a square may not have integral coordinates! (but **2 * coordinate** must be integer)

- Avoid double-counting if the center coincide with (x_0, y_0)

The End

- Any questions?