2026 Mini Competition 0 Editorial

2026-02-14

M2601
Love Color

M2601 Solution

Subtask 1: N = 1

- There is only one criterion.
- The number of valid components for red channel is $\max(R_{1,1}, R_{1,2}) - \min(R_{1,1}, R_{1,2}) + 1$.
- The total number of valid colors is the product of the number of valid component across all channels.

M2601 Solution

Subtask 2: $0 \le R_{i,1}, G_{i,1}, B_{i,1}, R_{i,2}, G_{i,2}, B_{i,2} \le 1$ for $1 \le i \le N$

- The total number of valid colors is at most 2 * 2 * 2 = 8.
- Enumerate all colors and check its validity.

M2601 Solution

Subtask 3: $R_{i,1} \le R_{i,2}$, $G_{i,1} \le G_{i,2}$, $B_{i,1} \le B_{i,2}$ for $1 \le i \le N$

- The criterion for i-th red channel is $R_{i,1} \le r \le R_{i,2}$.
- We can separate it into $R_{i,1} \le r$ and $r \le R_{i,2}$.
- To satisfy the first condition for all channels, r should be greater than or equal to $\max(R_{i,1})$
- To satisfy the second condition for all channels, r should be less than or equal to $\min(R_{i,2})$
- The total number of valid components is $\min(R_{i,2}) - \max(R_{i,1}) + 1$.
- When $\max(R_{i,1}) \ge \min(R_{i,2})$, the total number is 0.

M2601 Solution

Subtask 4: No additional constraints

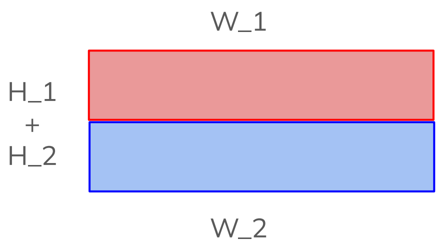
- Swap $R_{i,1}$ and $R_{i,2}$ when $R_{i,1} > R_{i,2}$.
- Repeat for green and blue channels.
- The question becomes subtask 3.

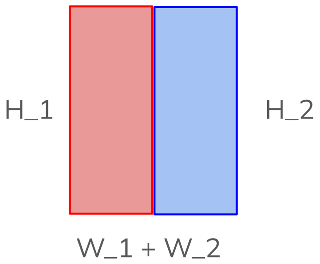
M2602
Perfect Collage

M2602 Solution

Subtask 1: N = 2

- There are two possible arrangements for a perfect collage:





- Which requires either W_1 = W_2 or H_1 = H_2
- Otherwise we output NO

M2602 Solution

Subtask 2: N = 3

- We can exhaust which photo to put in the top-left corner
- In order for the photos to form a perfect collage,
  - The remaining two photos should form a perfect collage
  - Furthermore, one of the resulting dimensions being equal to that of the first photo
- If a perfect collage cannot be formed after iterating all possibilities of the top left photo, output NO

M2602 Solution

Subtask 3: N = 4

- **Observation**: The area of the final collage must be the sum of the given photos
- If we are given the dimensions of the collage, we can also exhaust the photo to put in the bottom-right corner!
- Therefore, we can exhaust all of the following:
  - The dimensions of the collage
  - The photo to put in the top-left corner
  - The photo to put in the bottom-right corner
- We will end up with 3 different cases

M2602 Solution

Subtask 3: N = 4

- Case 0: The top-left photo and the bottom-right photo overlap
  - It is impossible for a perfect collage to be formed in this configuration
- Case 1: Both the top-right and bottom-left corners are filled
  - The remaining area should form a rectangle
  - Check if the remaining photos can fill that rectangle
- Case 2: At least one of top-right and bottom-left corner is not filled
  - For each unfilled corner, exhaust which photos to put in that respective corner
  - If only one corner was unfilled, check if the dimensions of the remaining area fits that of the last photo
  - Remember to check for overlaps!

M2603
Revolving Hearts

M2603 Solution

Observation 1: A heart will never overtake any other heart, since a heart will stop once there is a stopped heart in the next cell.

This means that we only have to care about whether the heart is

- still revolving;
- stopped by its own moving duration; or
- stopped by the heart right in front of it

at time T.

M2603 Solution

Observation 2: A heart with minimum moving duration must be either still revolving or stopped by its own moving duration at time T.

Let heart k be a heart with minimum moving duration. If D[k] > T , then heart k is still revolving at time T.
If D[k] ≤ T , then heart k has stopped moving since time D_k .

Let Y[i] be the final position of heart i.
Then, Y[k] = ((X[k] + min(D[k], T) – 1) mod M) + 1

M2603 Solution

Now we know how to determine whether a heart is still revolving or stopped by its own moving duration at time T.

The only thing left is to determine whether it is stopped by the next heart.

Note that we have also determined the final position of heart k, Y[k] .

We can use Y[k] to find Y[k−1], Y[k−2], ..., Y[1], Y[m], ..., Y[k+1] in order.

M2603 Solution

We can determine Y[i] if Y[i mod N +1] has been determined.

Note that min(D[i], T) is the number of moves that heart i will make if it is not affected by the next heart.

(Y[i mod N +1] – X[i] – 1 + M) mod M is the maximum number of moves heart i can make before it is stopped by heart (i mod N + 1).

So, Y[i] = ((X[i] + min(D[i], T, (Y[i mod N +1]–X[i]–1+M) mod M) – 1) mod M) + 1.

M2604
The Final Glimpse

M2604 Solution

Subtask 1: N = 1

- It is always not worse for Cupid to follow Psyche's direction (i.e. move from origin to P_1).
- Under such a path, the minimum distance is either 0 or attained when Cupid changes speed.
- Calculate at what times when Cupid changes speed and find out the positions of Cupid and Psyche accordingly.

M2604 Solution

Subtask 2: K = 1 / Subtask 3: N, K <= 100

- The direction that Cupid travels is no longer fixed as t varies.
- Intentionally small / weak test cases to water marks.
- Not very useful subtasks apart from being a safety net for full solutions.

M2604 Solution

Subtask 4: No additional constraints

- There are two types of “events”: Cupid changes speed, or Psyche changes direction. There are N + K events in total.
- Sort these events in increasing order of time that they happen, so that for the time interval between two consecutive events, the speed of Cupid and the velocity of Psyche are both fixed and known.
- For each such time interval, it simplifies to finding the minimum value of the expression $\mathrm{sqrt}((A + B * t)^2 + (C + D * t)^2) - (E + F * t)$ over $0 <= t <= T$, where A, B, C, D, E, F, T are constants (wrt the fixed speeds and velocities).

M2604 Solution

- Observe that the function f(t) = sqrt((A + Bt)^2 + (C + D * t)^2) - (E + F * t) is (strictly, provided that the appropriate quantities are nonzero) convex.
- So, we can ternary search on the minimum value of the expression over the interval 0 <= t <= T. (Alternatively, you can do some (vector) calculus to obtain the stationary point of f.)
- Remember to check for t = 0 and t = T too.

- One can achieve O(N + K) time complexity with some care, but other easier solutions with O((N + K) * (log(N + K) + log eps^{-1})) also fits in the time limit comfortably.

M2605
Valentine Messengers

M2605 Solution

Observation:

- Due to parity of a 2D grid, the possible destinations (x, y) each messenger reaches must satisfy:
  - $x + y = U_i + D_i + L_i + R_i \pmod 2$
  - This observation is true for all subtasks

M2605 Solution - Subtask 1

Subtask 1: N, M <= 2000, T_i = ‘UDLR’ or ‘DULR’ or ‘UDRL’

Do O(NM) checking:

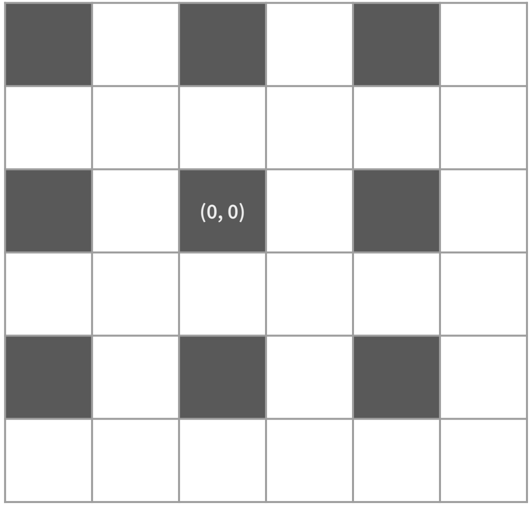
- T_i = ‘UDLR’
  - No misreading: just count how many lovers are at the destination
- T_i = ‘DULR’
  - LR is fixed, both U and D can be interpreted as U or D
  - Simply $U_i + D_i$ up or down movements from the origin
  - Vertical line of length = $U_i + D_i$, account for parity
- T_i = ‘UDRL’
  - Similar to ‘DULR’ except that it is a horizontal line

M2605 Solution - Subtask 2

Subtask 2: N, M <= 2000, T_i = ‘DURL’

Do O(NM) checking:

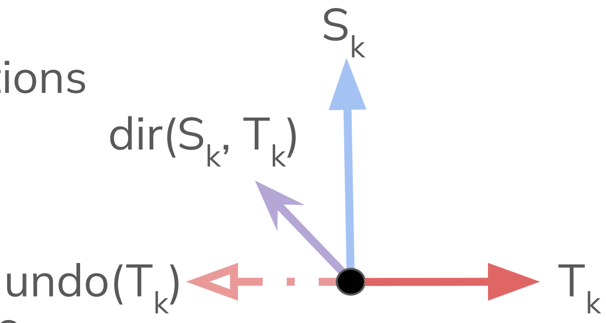
- Notice that we can consider U/D and L/R independently
- “Combine” horizontal and vertical lines from Subtask 1 to form a grid centered at the origin



M2605 Solution

Observation:

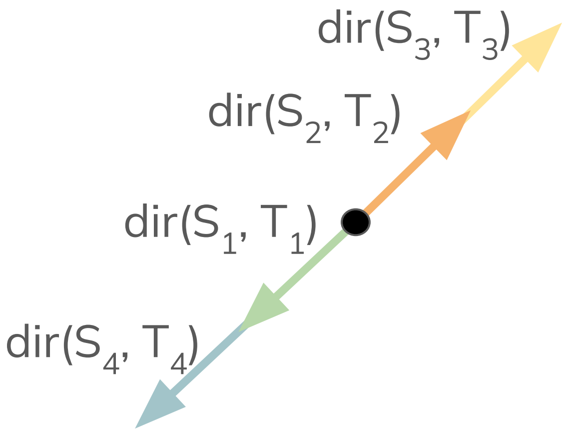
- Consider the original final destination where no directions are misread
- A misread is **undoing** a direction S_k and moving in direction T_k at the same time
- Using the original final destination as the new 'start' is much more useful than the origin



M2605 Solution - Subtask 3

Subtask 3: N, M <= 2000, ‘A’ misread as ‘B’ -> ‘B’ misread as ‘A’

- Subtask 1 and Subtask 2 covers all parallel directions misread as each other
- Consider non-parallel directions are misread as each other, for example T_i = ‘LRUD’
  - U misread as L -> undo U and do L -> DL
  - D misread as R -> undo D and do R -> UR
  - L misread as U -> undo L and do U -> UR
  - R misread as D -> undo R and do D -> DL
- All misreads are movements along the same diagonal

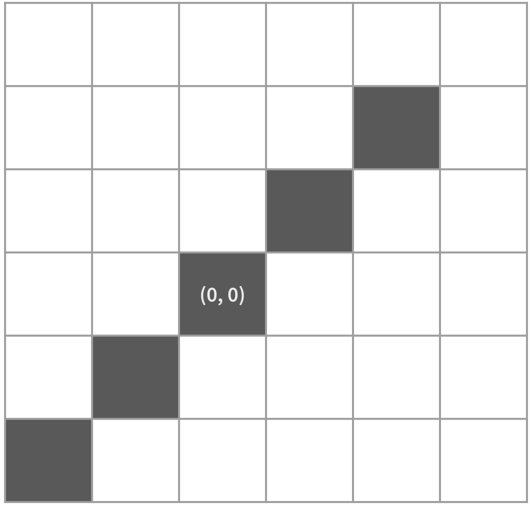


M2605 Solution - Subtask 3

In fact, the new cases to consider are all diagonal lines.

Do O(NM) checking:

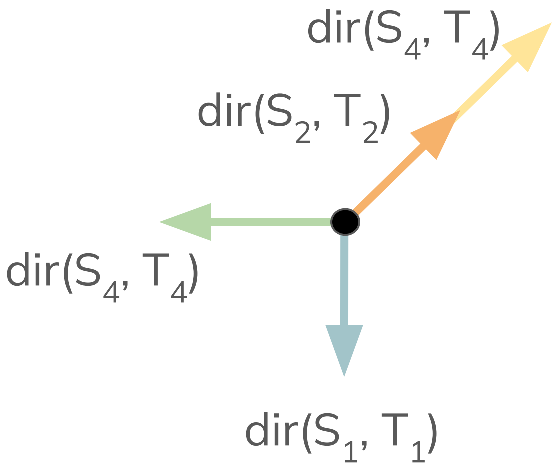
- Use previous subtask solutions
- Sum up the occurrences that contribute to the same diagonal direction
- Check if point lies on diagonal line



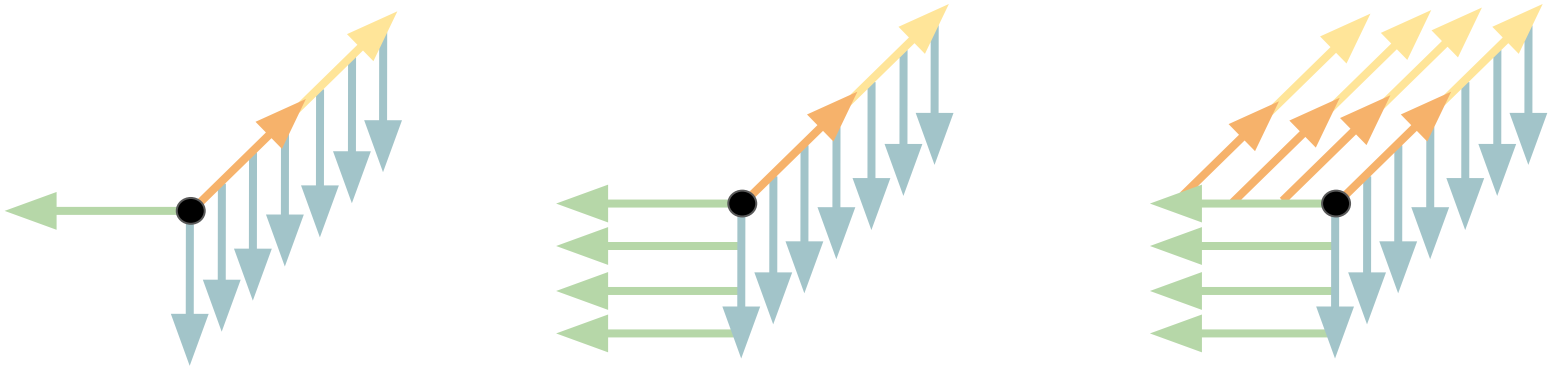
M2605 Solution - Subtask 4

Subtask 4: N, M <= 2000

- Taking inspiration from subtask 3, we consider each of S_k and T_k independently, summing up occurrences that contribute to the same directions
- Consider T = 'DRUL'



M2605 Solution - Subtask 4



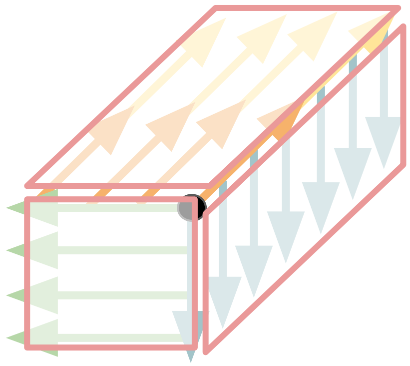
M2605 Solution - Subtask 4

Subtask 4: N, M <= 2000

- Take clockwise adjacent directions, each pair of adjacent directions forms a quadrilateral

Do O(NM) checking:

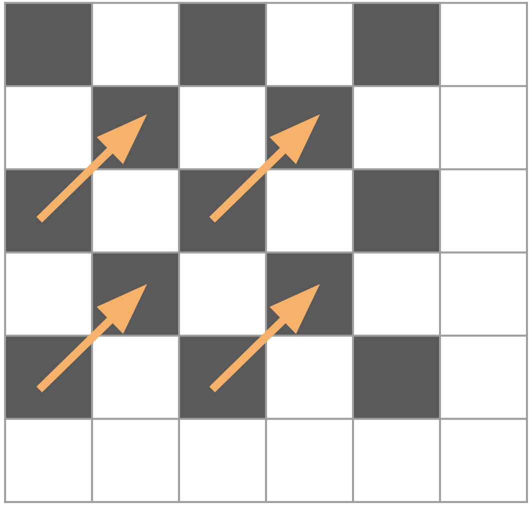
- Check whether each point lies within one of these quadrilaterals
  - Use x+y and x-y for diagonal axis comparisons



M2605 Solution - Subtask 4

Subtask 4: N, M <= 2000

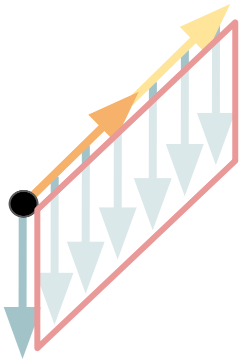
- Note that the rectangular case is different from Subtask 2
  - If there exists a diagonal move, will fill in the diagonal gaps
  - Appears if some occurrence of directions = 0



M2605 Solution - Subtask 5

Subtask 5: X_j, $Y_j \le 1000$, $0 \le U_i$, D_i, L_i, $R_i \le 1000$,
Exactly two of U_i, D_i, L_i, R_i is 0

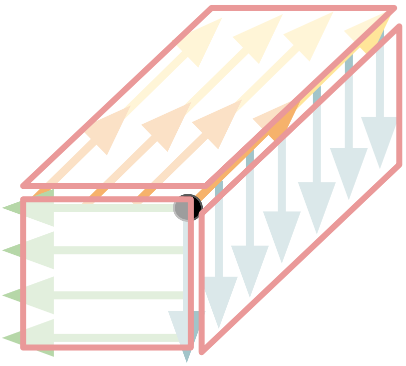
- At most 1 quadrilateral
- We can translate each quadrilateral into 2 constraints, count number of points satisfying 2 constraints -> 2D Partial Sum!
- 4 axis -> 4 x 3 / 2 = 6 pairs of axis -> Create 6 2D Partial Sum matrices (possibly 12 if you separate parity)



M2605 Solution - Subtask 6

Subtask 6: $X_j, Y_j \le 1000$, $0 \le U_i, D_i, L_i, R_i \le 1000$,

- Same with Subtask 5, but need to handle multiple quadrilaterals
  - May have double counting
  - Resize the quadrilateral or use inclusion-exclusion



M2605 Solution - Subtask 7

Full solution

- Grid is too large to use 2D Partial Sum

Instead of maintaining 2D Partial Sum:

- Sort 2D sum queries in ascending order
- Calculate queries with Sweep line + BIT (will be taught in training :))

Time complexity: $O((M + 10^5)\log(10^5))$

Note that the constant factor is quite large due to the pairs of possible dimensions