Problem E: Enemy Ships

Advanced Algorithms for Programming Contests

Restrictions

Time: 4 seconds
Memory: 1.2 GB

Problem description

The Milky Way is under attack by its closest neighbor, the Andromeda Galaxy. Thousands of andromedan space ships are invading, and the Milky Way's galactic council seems utterly overwhelmed by the situation. To give some strategic basis to their defense, and to enable trade ships to (relatively) safely continue their travels through the Milky Way, they have requested a program that keeps track of when and where enemy ships have been sighted.

The 2D map of the Milky Way they want to work with divides the galaxy into a 500×500 grid of quadratic cells, each of which denotes a *galactic sector*. The program should act on a timescale of days, reaching from 0 to 1000, and it should be able to quickly and adequately respond to queries of the following forms:

- t `SIGHTING` n x y
  signaling a sighting of n enemy ships in sector (x, y) on day t

- t `TRAVEL` p n x y r
  signaling that on day t a trade ship requests whether sector (x, y) is safe enough for it to go there – specifically, whether in the past p days (up to and including the current day t, so in days $t - p + 1, ..., t$) less than n enemy ships were sighted in the r-cube of sectors around (x, y) (i.e. in $[x - r, x + r] \times [y - r, y + r]$, where $r \geq 0$ is guaranteed); the output should be `YES` if it is safe and `NO` otherwise

- t `COUNTER` p x_1 y_1 x_2 y_2
  signaling that on day t a part of the Milky Way's defense troops is considering a counterattack that would disperse andromedan invaders from all sectors in the area $[x_1, x_2] \times [y_1, y_2]$ and wants to know exactly how many enemy ships have been sighted in said area in the past p days (see above)

Often times, the program will be asked to give out information about sightings on the current day (i.e. the day the request is made) long before all

sightings of that day have been registered by it. This problem is to be disregarded (i.e. the program should always reply instantly, based solely on the information it received up to the request).

Input

The input consists entirely of queries of the types described above, each printed in a separate line. All coordinates – including all corners of r-boxes in `TRAVEL` queries – are guaranteed to be integers lying on the map (reaching from 1 to 500 in both coordinate directions) and all query timestamps t are guaranteed to lie between 1 and 1000 (incl.). Also, no request will ask about sightings on days with indices smaller than 1 (i.e. $t - p + 1 \geq 1$), sightings will only ever report positive amounts of enemy ships not exceeding 1000 per sighting, and – of course – the queries will be sorted by the times t at which they are issued. The program is supposed to terminate when given the query

$$-1 \texttt{ SIGHTING } -1\ -1\ -1.$$

You may assume that there will be no more than 10^5 queries in a single execution of the program.

Output

For every non-terminating query of the types `TRAVEL` and `COUNTER`, output the correct result (as specified above) to that query on a separate line (in the order in which the queries are received).

Sample input and output

Input	**Output**
13 SIGHTING 332 5 8 22 TRAVEL 10 5 6 6 2 23 TRAVEL 10 5 6 6 2 37 SIGHTING 68 117 12 37 COUNTER 30 1 1 120 12 -1 SIGHTING -1 -1 -1	NO YES 400