

# Level 5 - Scale up



## Definitions

- We call the **map size** 'S'.
- The **map** is the area with x and y from -S to S inclusive.
- All the points in the map with integer coordinates are **grid points**.

## Task

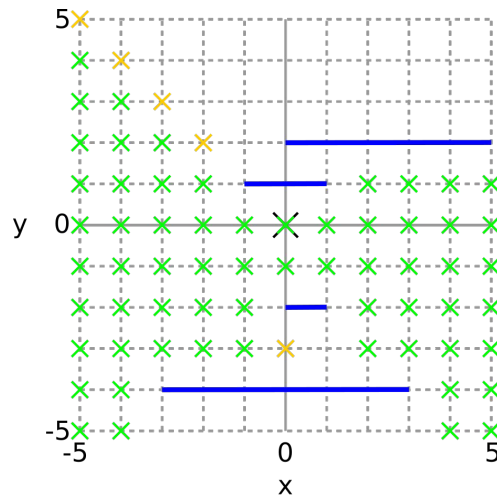
In order to optimize the hyperloop planning, we are now interested in the reachability of the entire map. That is, all grid points are now targets.




You will be provided with the map size S. Output the number of reachable grid points.

## Notes

Grid points on obstacles are unreachable.

Grid points separated from the start point only by the ends of obstacles may be considered reachable or unreachable. A range of answers will be accepted.

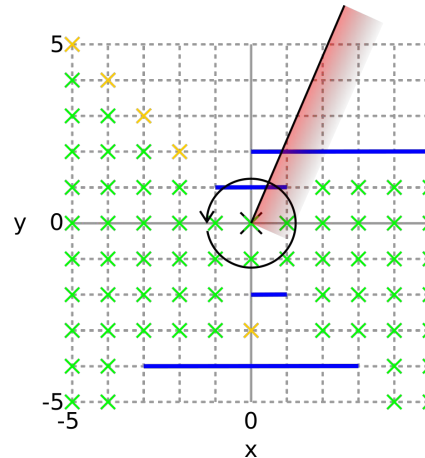


Legend	
	Obstacle
	Reachable grid point
	Grid point may be counted as reachable or unreachable

# Hints



Take note of the size of the  $S$  and  $V$  input parameters. The algorithm you developed for Level 4 may well be too slow to complete this task before the end of the contest. An efficient solution here is to use a ‘sweep-line’. This means we sort all the targets and obstacle ends by angle (relative to the start point) and iterate through them in this order, keeping track of which obstacles are intersected.



All commonly used programming languages have built-in functions for sorting arrays or lists with custom ordering. You shouldn't need to implement a sorting algorithm yourself.

# Data format

## Input

<S>

<V> the number of obstacles

V lines: <ObstacleX0> <ObstacleX1> <ObstacleY> an obstacle as before

## Output

<CountOfReachablePoints>



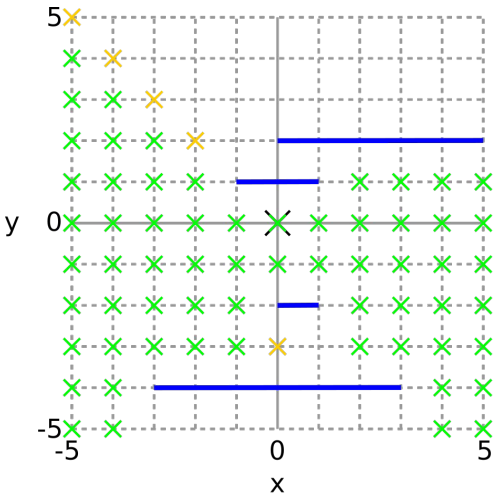
# Example

## Input

```
5
4
0 1 -2
-1 1 1
0 5 2
-3 3 -4
```

## Output

Any integer between 62 and 67 inclusive.



Legend

—

Obstacle

×

Reachable grid point

×

Grid point may be counted as reachable or unreachable

