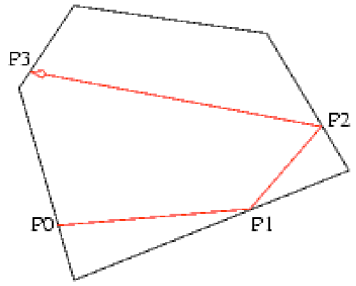
Fourth Southern African Regional
ACM Collegiate Programming Competition

Sponsored by IBM

Problem 4 – Purple balloon
Mirrors

You have been asked to design a security system for protecting the crown jewels. After careful consideration (and watching numerous Remington Steele episodes), you have decided to set up a laser tripwire alarm system. The system works by bouncing a laser beam off mirror strips fitted to the walls of the room. Since the beam will follow a fixed path through the room, you can place a single sensor at the end of the laser's path. If anyone breaks the beam at any point along the path that the laser follows, the light will not reach the sensor, setting off the alarm. The diagram below illustrates a how a beam originating at P0 is reflected off the walls, eventually reaching the point P3 (this is a top-down view of the room).



Your task is to calculate the path that the beam will take through the room so that you know where you have to place the sensor. Your input will be the geometry of the room, and the origin of the laser beam. Note that the problem domain is restricted to two dimensions.

The laser beam is reflected in a direction so that the angle of reflection (with respect to the wall) is equal to the angle of incidence. You don't have to worry about hitting corners; test cases will never have beams hitting the corner exactly.

Input

Your input consists of the following elements:

```
number_of_reflections
laser_x laser_y
laser_direction_x laser_direction_y
number_of_walls
wall_start_x wall_start_y wall_stop_x wall_stop_y
```

All the values are real numbers, except the number_of_reflections and number_of_walls parameters, which are integers.

You have to follow the path of the beam originating at position `(laser_x, laser_y)` travelling in direction `(laser_direction_x, laser_direction_y)`. Note that the laser direction is a vector, not a point.

Each problem will have `number_of_walls` wall specifications; only the first one is shown above. Your input may contain several problem specifications, each following the format defined above.

Output

Your output will be the coordinates (accurate to 2 decimal places) of the point where the beam strikes on its nth reflection, where n = `number_of_reflections`. For example, in the diagram above, your output will be the coordinates of point P3, with `number_of_reflections` = 3 (thus the endpoint is treated as part of the reflection count). Repeat this for every problem specified in the input.

Sample Input

```
5
0 0
1 0.5
4
-1 -1  1 -1
 1 -1  1  1
-1  1  1  1
-1  1 -1 -1
```

Sample Output

```
0.00 -1.00
```