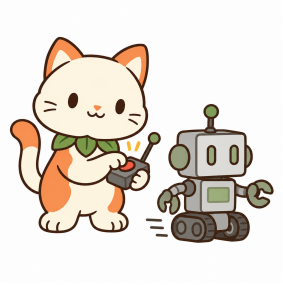
L Long Slides

Time limit: 2s

This problem has an attachment. You can use it to simulate and visualize the movements of the robot.



VegiCat has a robot on an infinite two-dimensional grid. He wants it to move from cell $s = (s_x, s_y)$ to cell $t = (t_x, t_y)$. Each cell has integer coordinates. The cells s and t may be the same.

The robot follows a string p of instructions. Each character is `L`, `R`, `U`, or `D`. Let $|p|$ denote the length of p.

The robot starts at s. Before it executes any instructions, you may place at most $|p|$ obstacles in distinct cells. The coordinates (x, y) of each obstacle must satisfy $-2 \cdot 10^9 \le x, y \le 2 \cdot 10^9$. Neither s nor t may contain an obstacle. Each obstacle occupies one cell and remains fixed throughout the process; all other cells are free of obstacles.

The robot executes the instructions $p_1, p_2, \ldots, p_{|p|}$ in order. For each instruction, starting from its current cell (x, y), it moves along the following sequence of cells:

- `L`: $(x, y) \to (x-1, y) \to (x-2, y) \to \cdots$;
- `R`: $(x, y) \to (x+1, y) \to (x+2, y) \to \cdots$;
- `U`: $(x, y) \to (x, y+1) \to (x, y+2) \to \cdots$;
- `D`: $(x, y) \to (x, y-1) \to (x, y-2) \to \cdots$.

The robot repeatedly takes one step in the specified direction while the next cell contains no obstacle. It stops in the cell immediately before the first obstacle in that direction. If the next cell in the specified direction already contains an obstacle before the instruction starts, the robot stays in place, but the instruction is still completed. If there is no obstacle ahead in that direction, the robot would move forever, and the placement is invalid.

The robot may reach t before all instructions are completed. It **must still execute all remaining instructions**. Construct any obstacle placement satisfying the requirements above such that the robot finishes at t after the last instruction, or report that no such placement exists.

Input

The first line contains an integer T ($1 \le T \le 10^5$), the number of test cases. The description of the test cases follows.

The first line of each test case contains two integers s_x and s_y ($-10^9 \le s_x, s_y \le 10^9$), the coordinates of the starting cell.

The second line contains two integers t_x and t_y ($-10^9 \le t_x, t_y \le 10^9$), the coordinates of the target cell.

The third line contains the instruction string p ($1 \le |p| \le 10^5$), consisting only of the characters `L`, `R`, `U`, and `D`.

It is guaranteed that the sum of $|p|$ over all test cases does not exceed 10^5.

Output

For each test case, output an answer in input order. Obstacle placements for different test cases are independent.

If no valid placement exists, output `NO` on a line by itself.

Otherwise, output `YES` on the first line. On the second line, output an integer k ($0 \le k \le |p|$), the number of obstacles.

Then output k lines. The i-th of these lines must contain two integers x_i and y_i ($-2 \cdot 10^9 \le x_i, y_i \le 2 \cdot 10^9$), specifying an obstacle in cell (x_i, y_i).

All k coordinate pairs must be distinct, and none may equal s or t. The placement must allow every instruction to finish and leave the robot at t after the last instruction. If there are multiple solutions, output any of them.

You can output the yes or no verdict in any case. For example, `yEs`, `yes`, `Yes`, and `YES` are all accepted as positive responses.

Sample Input 1	Sample Output 1
6 0 0 1 0 RL 0 0 -1 0 URD 0 0 8 0 RLR 0 0 0 0 UDUD 0 0 2 5 RDULR 0 0 1 1 RURURU	NO NO YES 2 -1 0 9 0 YES 2 0 -1 0 1 YES 5 -1 5 2 -1 2 6 3 0 3 5 YES 3 1 2 2 0 2 1