PROBLEM 9 – POLYGONAL LINES

A rectangle is to be cut by a sequence of one or more straight line segments joining a start node to an end node, both on the rectangle border.

Write a program that will read in details of the rectangle and the dividing line and determine whether the cut produces exactly two parts that could slide apart while remaining in the same plane.

The following sample diagrams contain:

1. several cases where the answer is "`Yes`", i.e., the cutting line produces exactly two parts that can slide away as required (**1** - **5**), and
2. several cases where the answer is "`No`", i.e., the cutting line doesn't produce exactly two parts as required (**6** - **10**).

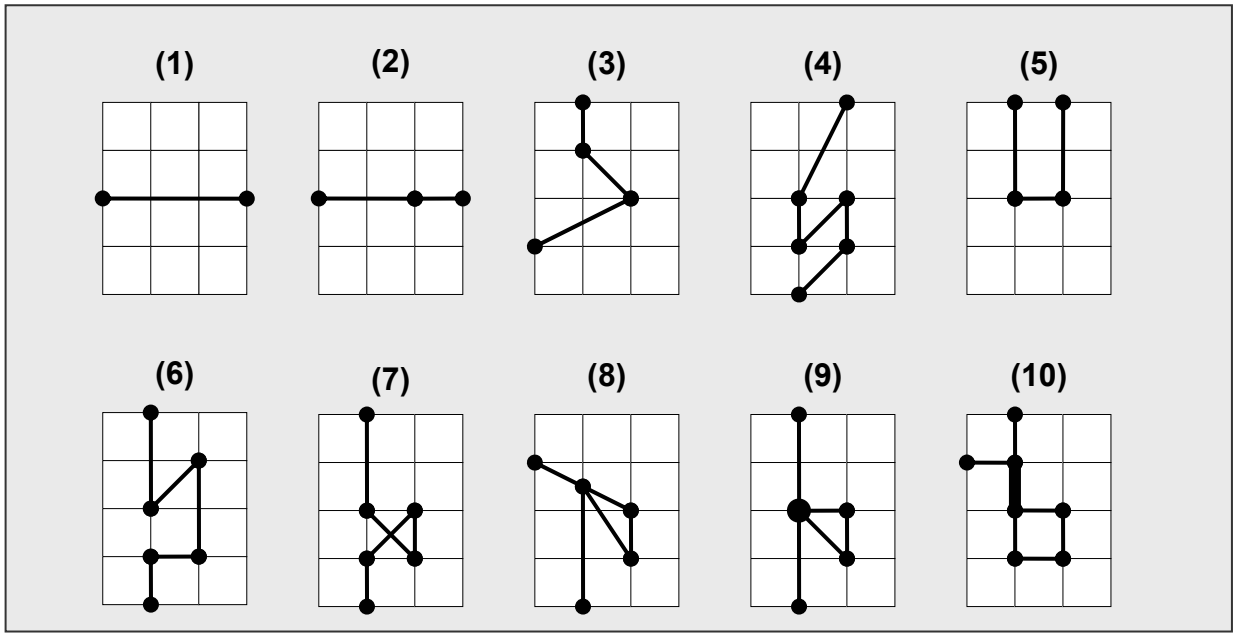

Diagram (**6**) shows two interlocked parts that cannot slide apart. Diagrams (**7** – **10**) show cuttings that produce three parts instead of the required two. Diagram (**7**) shows two intersecting segments. Diagram (**8**) shows a node that touches another segment. Diagram (**9**) shows two duplicate nodes that overlap (on the thicker spot). Diagram (**10**) shows two segments that overlap (along the thicker line). This overlapping will be more obvious in the sample input section below (where the grid size is assumed to be 10).

The programmer that initially received this task noticed that if the two parts can slide apart then they can always slide apart along the slope of at least one of the given line segments. However, he was unable to put this idea to work. Your task is to help him and write the required program.

INPUT FORMAT

Input consists of a number of scenarios. Each scenario starts with a "title" line containing a scenario title followed by three integers X_{MAX}, Y_{MAX}, and N, separated by single spaces. The scenario title is a string of 1 to 20 letters, digits, and underscores (with no intervening spaces). It is assumed that our rectangle is aligned with the axes with the origin at the bottom-left corner, and that X_{MAX} and Y_{MAX} specify the top-right corner, where $10 \le X_{MAX}, Y_{MAX} \le 1{,}000{,}000$. N represents the number of nodes of the cutting line, $2 \le N \le 200{,}000$. A single # on a line indicates the end of input.

This "title" line is followed by one or more lines, as needed to describe all nodes of the cutting line, in succession, 0, 1, 2, …, N-1. Each "nodes" line contains one or more pairs of non-negative integers, each giving the x and y coordinates of the corresponding node.

You can assume that:

- the start and the end nodes are distinct and lie on the rectangle borders,
- all other nodes lie within the interior area of the rectangle,
- there are no successive duplicate nodes (i.e., no 0 length segments),
- there are no successive overlapping segments (such as "0 0 4 4 2 2").

SAMPLE INPUT:

```
Case_1 30 40 2
0 20 30 20
Case_2 30 40 3
30 20 20 20 0 20
Case_3 30 40 4
0 10 20 20 10 30 10 40
Case_4 30 40 6
20 40 10 20 10 10
20 20 20 10 10 0
Case_5 30 40 4
20 40 20 20 10 20 10 40
Case_6 30 40 6
10 0 10 10 20 10 20 30 10 20 10 40
Case_7 30 40 6
10 0 10 10 20 20 20 10 10 20 10 40
Case_8 30 40 5
10 0 10 25 20 10 20 20 0 30
Case_9 30 40 6
10 0 10 20 20 10 20 20 10 20 10 40
Case_10 30 40 7
10 40 10 20 20 20 20 10 10 10 10 30 0 30
#
```

OUTPUT FORMAT

Output consists of one line for each scenario. There are two cases:

1. The cutting line meets all our requirements and produces exactly two parts that can slide apart. In this case assume that ((*x1*, *y1*), (*x2*, *y2*)) is the first of the line segments whose slope can be used for sliding apart the two parts. Output the scenario title, followed by a colon (“`:`”), a space (“ ”), the word “`Yes`”, another space (“ ”), and then the four integers *x1*, *y1*, *x2*, *y2*, separated by single spaces (these coordinates must appear in their input sequence).

2. Otherwise output the scenario title, followed by a colon (“`:`”), a space (“ ”), and the word “`No`”.

SAMPLE OUTPUT:

```
Case_1: Yes 0 20 30 20
Case_2: Yes 30 20 20 20
Case_3: Yes 0 10 20 20
Case_4: Yes 10 10 20 20
Case_5: Yes 20 40 20 20
Case_6: No
Case_7: No
Case_8: No
Case_9: No
Case_10: No
```