TOYS

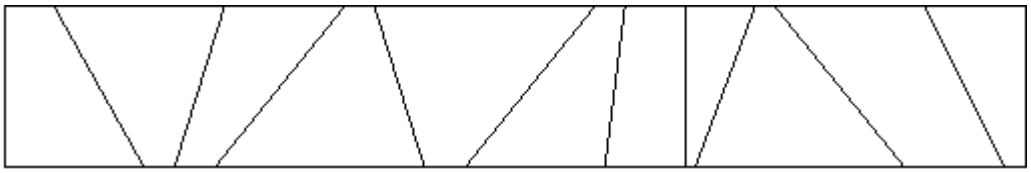
Time Limit: 2000MS **Memory Limit:** 65536K

Total Submissions: 3463 **Accepted:** 1691

Description

Calculate the number of toys that land in each bin of a partitioned toy box.
Mom and dad have a problem - their child John never puts his toys away when he is finished playing with them. They gave John a rectangular box to put his toys in, but John is rebellious and obeys his parents by simply throwing his toys into the box. All the toys get mixed up, and it is impossible for John to find his favorite toys.

John's parents came up with the following idea. They put cardboard partitions into the box. Even if John keeps throwing his toys into the box, at least toys that get thrown into different bins stay separated. The following diagram shows a top view of an example toy box.



For this problem, you are asked to determine how many toys fall into each partition as John throws them into the toy box.

Input

The input file contains one or more problems. The first line of a problem consists of six integers, n m x1 y1 x2 y2. The number of cardboard partitions is n ($0 < n <= 5000$) and the number of toys is m ($0 < m <= 5000$). The coordinates of the upper-left corner and the lower-right corner of the box are (x1,y1) and (x2,y2), respectively. The following n lines contain two integers per line, Ui Li, indicating that the ends of the i-th cardboard partition is at the coordinates (Ui,y1) and (Li,y2). You may assume that the cardboard partitions do not intersect each other and that they are specified in sorted order from left to right. The next m lines contain two integers per line, Xj Yj specifying where the j-th toy has landed in the box. The order of the toy locations is random. You may assume that no toy will land exactly on a cardboard partition or outside the boundary of the box. The input is terminated by a line consisting of a single 0.

Output

The output for each problem will be one line for each separate bin in the toy box. For each bin, print its bin number, followed by a colon and one space, followed by the number of toys thrown into that

bin. Bins are numbered from 0 (the leftmost bin) to n (the rightmost bin). Separate the output of different problems by a single blank line.

Sample Input

```
5 6 0 10 60 0
3 1
4 3
6 8
10 10
15 30
1 5
2 1
2 8
5 5
40 10
7 9
4 10 0 10 100 0
20 20
40 40
60 60
80 80
 5 10
15 10
25 10
35 10
45 10
55 10
65 10
75 10
85 10
95 10
0
```

Sample Output

```
0: 2
1: 1
2: 1
```

3: 1
4: 0
5: 1

0: 2
1: 2
2: 2
3: 2
4: 2

Hint

As the example illustrates, toys that fall on the boundary of the box are "in" the box.

Source

Rocky Mountain 2003

Intersecting Lines (pku 1269)

Time Limit: 1000MS **Memory Limit:** 10000K

Total Submissions: 3589 **Accepted:** 1761

Description

We all know that a pair of distinct points on a plane defines a line and that a pair of lines on a plane will intersect in one of three ways: 1) no intersection because they are parallel, 2) intersect in a line because they are on top of one another (i.e. they are the same line), 3) intersect in a point. In this problem you will use your algebraic knowledge to create a program that determines how and where two lines intersect.

Your program will repeatedly read in four points that define two lines in the x-y plane and determine how and where the lines intersect. All numbers required by this problem will be reasonable, say between -1000 and 1000.

Input

The first line contains an integer N between 1 and 10 describing how many pairs of lines are represented. The next N lines will each contain eight integers. These integers represent the coordinates of four points on the plane in the order x1y1x2y2x3y3x4y4. Thus each of these input lines represents two lines on the plane: the line through (x1,y1) and (x2,y2) and the line through (x3,y3) and (x4,y4). The point (x1,y1) is always distinct from (x2,y2). Likewise with (x3,y3) and (x4,y4).

Output

There should be N+2 lines of output. The first line of output should read INTERSECTING LINES OUTPUT. There will then be one line of output for each pair of planar lines represented by a line of input, describing how the lines intersect: none, line, or point. If the intersection is a point then your program should output the x and y coordinates of the point, correct to two decimal places. The final line of output should read "END OF OUTPUT".

Sample Input

```
5
0 0 4 4 0 4 4 0
5 0 7 6 1 0 2 3
5 0 7 6 3 -6 4 -3
2 0 2 27 1 5 18 5
0 3 4 0 1 2 2 5
```

Sample Output

```
INTERSECTING LINES OUTPUT
POINT 2.00 2.00
NONE
LINE
POINT 2.00 5.00
POINT 1.07 2.20
END OF OUTPUT
```

Source

Mid-Atlantic 1996

Pick-up sticks

Time Limit: 3000MS **Memory Limit:** 65536K

Total Submissions: 3435 **Accepted:** 1209

Description

Stan has n sticks of various length. He throws them one at a time on the floor in a random way. After finishing throwing, Stan tries to find the top sticks, that is these sticks such that there is no stick on top of them. Stan has noticed that the last thrown stick is always on top but he wants to know all the sticks that are on top. Stan sticks are very, very thin such that their thickness can be neglected.

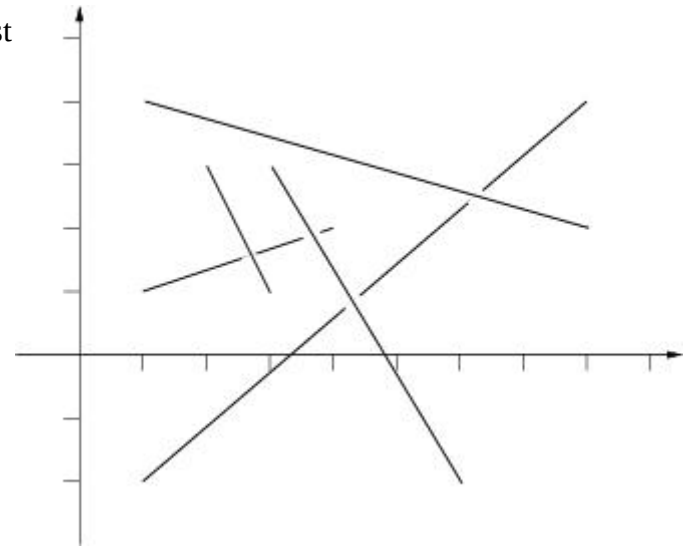
Input

Input consists of a number of cases. The data for each case start with 1 <= n <= 100000, the number of sticks for this case. The following n lines contain four numbers each, these numbers are the planar coordinates of the endpoints of one stick. The sticks are listed in the order in which Stan has thrown them. You may assume that there are no more than 1000 top sticks. The input is ended by the case with n=0. This case should not be processed.

Output

For each input case, print one line of output listing the top sticks in the format given in the sample. The top sticks should be listed in order in which they were thrown.

The picture to the right below illustrates the first case from input.



Sample Input

```
5
1 1 4 2
2 3 3 1
1 -2.0 8 4
1 4 8 2
3 3 6 -2.0
3
0 0 1 1
1 0 2 1
2 0 3 1
0
```

Sample Output

Top sticks: 2, 4, 5.
Top sticks: 1, 2, 3.

Hint

Huge input,scanf is recommended.

Source

Waterloo local 2005.09.17

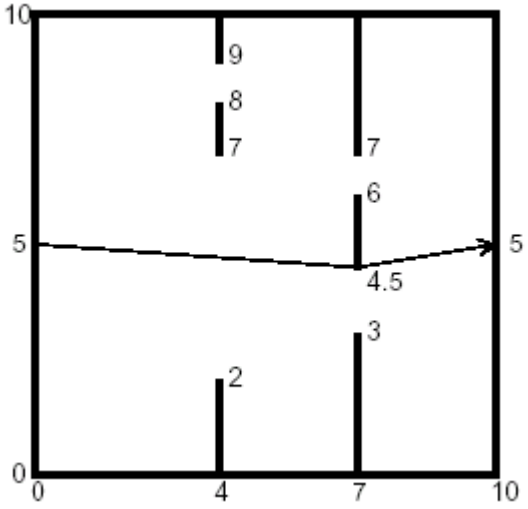
The Doors (pku 1556)

Time Limit: 1000MS **Memory Limit:** 10000K

Total Submissions: 1937 **Accepted:** 884

Description

You are to find the length of the shortest path through a chamber containing obstructing walls. The chamber will always have sides at x = 0, x = 10, y = 0, and y = 10. The initial and final points of the path are always (0, 5) and (10, 5). There will also be from 0 to 18 vertical walls inside the chamber, each with two doorways. The figure below illustrates such a chamber and also shows the path of minimal length.



Input

The input data for the illustrated chamber would appear as follows.

2
4 2 7 8 9
7 3 4.5 6 7

The first line contains the number of interior walls. Then there is a line for each such wall, containing five real numbers. The first number is the x coordinate of the wall (0 < x < 10), and the remaining four are the y coordinates of the ends of the doorways in that wall. The x coordinates of the walls are in increasing order, and within each line the y coordinates are in increasing order. The input file will contain at least one such set of data. The end of the data comes when the number of walls is -1.

Output

The output should contain one line of output for each chamber. The line should contain the minimal path length rounded to two decimal places past the decimal point, and always showing the two decimal places past the decimal point. The line should contain no blanks.

Sample Input

```
1
5 4 6 7 8
2
4 2 7 8 9
7 3 4.5 6 7
-1
```

Sample Output

```
10.00
10.06
```

Source

Mid-Central USA 1996