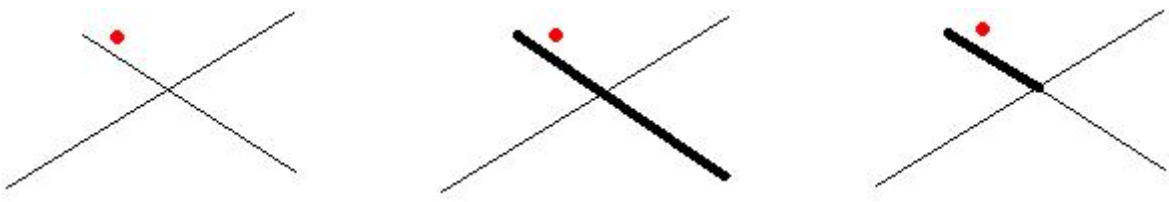
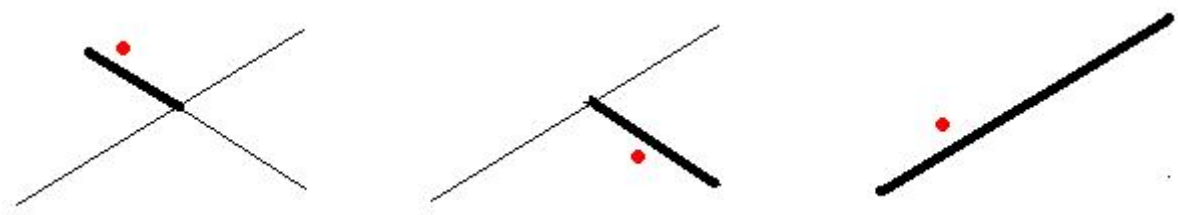
Google SketchUp is an easy-to-use program that lets you create, modify and share 3D models. In this problem, you're to write a simplified version of SketchUp called My SketchUp. Since 3D is complex, My SketchUp will be in 2D only.

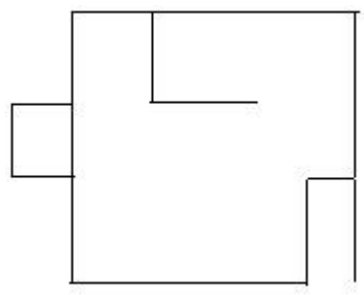
SketchUp is intuitive. To understand this, suppose you've drawn two line segments that intersect other, like this:



Then, you click your mouse at the red dot. If you're using AutoCAD, you'll select a whole segment (shown in the middle picture), but in My SketchUp, you'll only select a small segment (shown in the right picture), because the two segment you've drawn cut each other! What's more, if you remove two small segments as shown below, the other two small segments will automatically join up to become one segment again! Note that in the middle picture, you cannot select the "long" segment as a whole, because it's still cut into two pieces.



Here is the general rule: segments that intersect each other actually cut each other; and there will be no "redundant" points that can be removed without affecting the appearance of the picture. As a result, the in-memory data structure of the segments can be deduced merely from the appearance. **If two pictures look the same, they are the same internally**. For example, if you draw (0,0)-(1,0), then (1,0)-(2,0), you'll have only one segment: (0,0)-(2,0), if you draw (0,0)-(1,0) twice, you'll only get one. In the picture below, there are 14 vertices and 15 segments (no matter how you draw this picture!!).



Your task is to execute a sequence of commands (described in the input format section) and print the description of the resulting picture. Vertices are sorted in ascending order of x, then ascending order of y; Segments are represented by a pair of integers a and b ($a < b$), that means the segment is connecting vertex a and vertex b (vertices are numbered from 1).

Input

There will be at most 25 test cases. Each case begins with one integer n ($1 \le n \le 100$), the number of operations. Each of the following n lines is formatted as one of:

`DRAW` x_1 y_1 x_2 y_2 x_3 y_3 ...
or:

`REMOVE` x y d

In the *draw* operation, you're drawing a poly-line $(x_1, y_1) - (x_2, y_2) - (x_3, y_3) - ...$, note that if the last point equals to the first point, you're actually drawing closed poly-line (but not necessarily a polygon, since the poly-line could be self-intersecting. There will be at least 2 and at most 20 points in a draw operation.

In the *remove* operation, all the line segments whose distance from (x, y) is at most d, are removed *simultaneously* (be careful about this!). If no segments satisfy this condition, this operation takes no effect. $-1000 \le x_1, y_1, x_2, y_2 \le 1000$, $0 \le d \le 10$.

There is a special command '`END`' following the last draw/remove operation. The last test case is followed by a line with $n = 0$, which should not be processed.

Output

For each test case, print the number of vertices, followed by the coordinates of the vertices (one vertex per line), sorted as stated in the problem statement. The next line contains the number of segments, followed by the descriptions of the segments.

Tips: In this problem, your output must match the standard output perfectly. In order to prevent you from printing '`-0.00`' instead of '`0.00`', you're encouraged to add to small number (e.g. 1e-6) to each number you print.

Sample Input

```
7
DRAW 10 0 50 0 50 30 10 30 10 0
DRAW 0 10 10 10 10 20 0 20 0 10
DRAW 40 0 40 10 50 10 50 0
REMOVE 45 1 1
DRAW 50 20 20 20 20 30
DRAW 40 20 40 30
REMOVE 42 23 3
END
7
DRAW 0 0 10 0
DRAW 1 0 11 0
DRAW 12 0 15 0
DRAW 5 0 5 1
DRAW 8 0 8 1
REMOVE 5 2 1
DRAW 11 0 12 0
END
0
```

Sample Output

```
14
0.00 10.00
0.00 20.00
10.00 0.00
10.00 10.00
10.00 20.00
10.00 30.00
20.00 20.00
20.00 30.00
40.00 0.00
40.00 10.00
40.00 20.00
50.00 0.00
50.00 10.00
50.00 30.00
15
1 2
1 4
2 5
3 4
3 9
4 5
5 6
6 8
7 8
7 11
8 14
9 10
10 13
12 13
13 14
4
0.00 0.00
8.00 0.00
8.00 1.00
15.00 0.00
3
1 2
2 3
2 4
```