The first formal axiom list for origami was published by Humiaki Huzita and Benedetto Scimemi and has come to be known as the Huzita axioms. These axioms describe the ways in which a fold line can be generated by the alignment of points and lines. A version of the six axioms follows.

1. For points p_1 and p_2 , there is a unique fold that passes through both of them.
2. For points p_1 and p_2, there is a unique fold that places p_1 onto p_2.
3. For lines l_1 and l_2, there is a fold that places l_1 onto l_2.
4. For a point p_1 and a line l_1, there is a unique fold perpendicular to l_1 that passes through point p_1.
5. For points p_1 and p_2 and a line l_1, there is a fold that places p_1 onto l_1 and passes through p_2.
6. For points p_1 and p_2 and lines l_1 and l_2, there is a fold that places p_1 onto l_1 and p_2 onto l_2.

Roman is a good coder, but he is new to origami construction, so he decided to write a program to calculate the necessary folds for him. He already finished coding the cases for the first five axioms, but now he is stuck on the harder case, the axiom number 6. So he decided to hire a team of good coders — your team — to implement this case for his program.

Input

The input consists of one or more test cases. The total number of test cases t is specified in the first line of the input file.

Each test case consists of exactly four lines, describing l_1, p_1, l_2 and p_2, in that order. Each line is described by four integers — the coordinates of two different points lying on it: x_1, y_1, x_2, y_2. Each point is described just by two integers — its x and y coordinates. All coordinates do not exceed 10 by their absolute values. It is guaranteed that neither p_1 lies on l_1 nor p_2 lies on l_2. Lines l_1 and l_2 are different, but points p_1 and p_2 may be the same.

Output

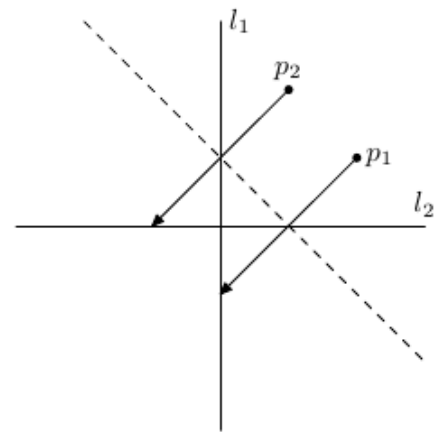
For each test case write a separate line with the description of the straight line one should use for folding.

Use the same format as in the input — specify the coordinates of two points on it. Either x or y coordinates of those two points must differ by at least 10^{-4} . Coordinates must not exceed 10^9 by their absolute value. The judging program will check that both the distance between p_1 after folding and l_1 ; and the distance between p_2 after folding and l_2 do not exceed 10^{-4}. If there are multiple solutions, any one will do. If there are no solutions, output the line of four zeros, separated by spaces.

Note: The picture below illustrates the first sample. The fold line is dashed.

Sample Input

```
2
0 0 0 1
2 1
0 0 1 0
1 2
0 0 0 1
5 0
1 0 1 1
6 0
```



Sample Output

```
0.0 1.0 2.0 -1.0
0 0 0 0
```