2D Matrix

CPSPC 2010 Prague, Den 1.

Submit:	`matrix.c` / `matrix.cpp` / `matrix.pas`	Time limit:	2 s
Input:	stdin	Memory limit:	64 MB
Output:	stdout	Points:	100

Our story takes places in the times, when Matrix did not have graphics card which would be powerful enough to create 3D illusion and therefore the people should be satisfied with 2D world. And in this 2D world, it was necessary to model the universe. Such universe consits from galaxies and galaxies consist from stars. Unfortunately, Matrix did not have too much memory, so in fact there were only two galaxies. And for those two galaxies, Matrix spent its last memory and thus it cannot remember for stars to which galaxy they belong. And that is the reason why it does the following trick. Both galaxies are centrally symmetric. Then it is possible to calculate to which galaxy every star belongs.

Task:

Write a program, that splits the given set of points in the plane into two disjoint sets such that both sets are centrally symmetric. If there are more such divisions print all of them. If there is not such division, something went terribly wrong and you have only to print "MATRIX PANIC".

Input:

On the first line of the input, there is the integer number N ($N \leq 150$) followed by N lines. Every line then contains two integers X_i and Y_i ($-10^9 \leq X_i, Y_i \leq 10^9$) which stand for the coordinates of the i-th star. You can assume, that the given set is not centrally symmetric and therefore both galaxies are nonempty.

Output:

The output consists of the several lines. The number of lines is equal to the number of solutions. Each line stands for one solution and contains four numbers. The first two of them stand for the coordinates of the centre of the first galaxy and the second two stand for the coordinates of the centre of the second galaxy. The coordinates have to be written precisely with all decimal digits without trailling zeroes. If there does not exist any division into two sets, write only the message "MATRIX PANIC".

Example:

input	output
6 0 2 2 0 2 2 0 0 4 0 4 1	1 1 4 0.5 2 0.5 2 1 2 1 2 0.5 4 0.5 1 1