Rooks

Consider a square board with *N* rows (called ranks) and *N* columns (called files). *K* of the squares are blocked by obstacles. Pieces similar to the rooks in chess are placed on this board. Two rooks are said to be attacking each other if they are on the same rank or file and there are no obstacles between them.

Task

Given a positive integer *N* and the positions of the *K* obstacles, place as many rooks as possible on the board so that no two rooks attack each other.

Input data

The first line of the input file `rooks.in` contains *N* and *K*, separated by a space. Each of the following *K* lines contain a pair of numbers *r* and *f*, separated by a space describing the rank and file of an obstacle. All the obstacles are distinct.

Output data

The first line of the output file `rooks.out` must contain a single number *S*, the highest number of non-attacking rooks that the table can accommodate. Each of the following *S* lines must contain a pair of numbers *r* and *f*, separated by a space, showing the rank and file of a rook. Any correct placement of S rooks will be accepted.

Limits and constraints

- $1 \leq N \leq 1,000$
- $1 \leq K \leq \min(N^2; 2,000)$
- Ranks and files are numbered from 1 to *N*.
- Time limit: 0.4 seconds
- Memory limit: 64 MB

Example

rooks.in	rooks.out	Explanation
5 2 3 2 2 4	7 1 4 2 2 2 5 3 1 3 4 4 3 5 2	