2006 Canadian Computing Competition
Day 2, Question 2

Input file: `domino.in`
Output file: `domino.out`
Source file: `n:\domino\domino.___`

Crazy Cuckoo Charlie's Dominoes

Crazy Cuckoo Charlie is a Computer Science student trying to get himself into the Book of Waterloo Records. Unfortunately, he has no special skills. He only has a lot of time, and a lot of dominoes. He's going to try to break the record for "Waterloo's Longest Domino Chain", which involves putting as many dominoes end-to-end as possible.

Dominoes have two halves. Each half has a number of dots on it. To break the Waterloo Record, Charlie must place all the dominoes end-to-end in a line. To make it more difficult, however, the Book of Waterloo Records editors have added an extra rule: the adjacent halves of each pair of neighbouring dominoes must share the same number of dots, as in the picture above.

Charlie wants to use all his dominoes when making the chain. But due to this rule, Charlie won't be able to make a chain using all his dominoes (one trivial example is "1 2" and "4 5"). He's willing to go out and buy extra dominoes to complete his chain. Obviously he wants to buy as few dominoes as possible, to save money for more important things like pizza and pop.

Can you help Charlie figure out the minimum number of dominoes required to complete his chain?

Hint

As you learned this week, you can model a lot of problems with graphs. This is one such problem.

By the way, an *Euler path* is a path which traverses all edges of a graph. (This is different than a Hamiltonian path. Very different.) A graph has an Euler path if and only if two conditions hold:

1. The graph is connected (you can get from anywhere to anywhere).
2. The count of verticies with *odd vertex degree* (the number edges touching the vertex is odd, not even) is at most two.

Input

Input consists of a series of test cases in a single file. Each test case describes one possible domino collection. The first line contains an integer N indicating the number of test cases in the file. Each test case begins a line with the integer K, $1 \leq K \leq 10000$, the number of dominoes Charlie has. Each of the next K lines describes a domino in his collection, in the form "A B", $0 \leq A \leq B \leq 50000$, where A and B are positive integers indicating the number of dots on each half of the domino. Don't forget you can flip dominoes around!

Output

For each test case, output on a line, the number X, where X is the minimum number of dominoes Charlie needs to buy in order to be able to make a chain with all his dominoes. On the next X lines should be a list of the dominoes Charlie needs, in the form "A B". Print the list of dominoes in ascending lexicographical order (from smallest to biggest). If there are multiple sets of dominoes (all of the minimum size) that Charlie could buy, pick the lexicographically least one.

To determine which domino is lexicographically least we use the following check: For two given dominoes "A B", where $A \leq B$, we pick the domino with the lower A. If both dominoes have the same A we pick the domino with the lower B.

Sample Input

```
2
4
1 10
2 8
8 10
4 606
6
3 8
3 5
1 3
1 5
1 8
5 8
```

Output for Sample Input

```
1
1 4
1
1 3
```