Task: KON
Conductor

XXV OI, Stage II, Day one. Source file `kon.*` Available memory: 256 MB. *14.02.2018*

Byteasar is a conductor on the most popular train connection in Byteotia. There are m stops along his route, conveniently numbered from 1 to m. As the passengers may enter or leave the train at each stop, Byteasar should inspect the tickets between every pair of successive stops to make sure that everyone has a valid ticket. However, this would take too much time, as a conductor has other duties besides inspecting tickets.

Byteasar is determined to resolve this conundrum methodically. He has determined n most popular, i.e., most frequently traveled, segments of his route. He denotes each such segment by a pair a_i, b_i, where a_i is the number of the stop where the passenger enters the train and b_i the number of the stop where they leave the train. Byteasar would like to inspect the tickets the minimum number of times while ensuring that everyone traveling along one of these n popular segments will have their ticket inspected at least once. I.e., for each segment a_i, b_i, there has to be at least one inspection between the stops a_i and b_i. For practical reasons, we assume that the ticket inspection may not take place while the train is stopped at a station.

Moreover, Byteasar realizes that he has to alter the inspection times, for otherwise frequent commuters would quickly notice his pattern and could modify their routes so as to avoid the control. Hence, Byteasar is keen to know all the minimum inspection schedules, where two schedules differ if there is a pair of successive stops such that a ticket control is scheduled between them in exactly one of the two schedules. In the fact, only the remainder modulo 1 000 000 007 of the number of distinct minimum inspection schedules will suffice for Byteasar at the moment.

Input

The first line of the standard input contains a single integer $z \geq 1$ that specifies the number of unit tests to be solved. Descriptions of these unit tests follow, each formatted as follows.

In the first line of a description there are two integers m and n ($1 \leq m \leq 10^9$, $1 \leq n$), separated by a single space, which specify the number of stops and the number of popular segments respectively. Then n lines follow, describing the segments: The i-th such line contains two integers a_i, b_i ($1 \leq a_i < b_i \leq m$), separated by a single space; these mean that the i-th popular segment starts at the stop no. a_i and ends at the stop no. b_i. Every ordered pair (a_i, b_i) will appear on input at most once (within a single test set).

Output

Exactly z lines, containing the answers to successive test sets, should be printed to the standard output. The i-th line should contain two integers separated by a single space – the minimum number of ticket inspections to satisfy Byteasar's requirements and the number of distinct minimum sets of such inspections modulo 1 000 000 007.

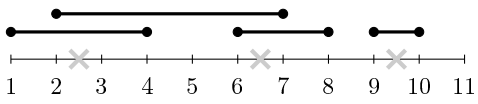
Example

For the following input data:

```
2
11 4
1 4
6 8
2 7
9 10
3 2
1 2
2 3
```

the correct answer is:

```
3 5
2 1
```



Explanation for the example: In the first unit test, three ticket inspections are required to cover all four popular segments. The figure depicts one possible control schedule: after the train leaves the stations $\{2, 6, 9\}$. The remaining control schedules are: $\{2, 7, 9\}$, $\{3, 6, 9\}$, $\{3, 7, 9\}$, and $\{1, 6, 9\}$; there are five minimum control schedules in total.

Sample grading tests:

`1ocen:` $n = 4$, $m = 10$;

`2ocen:` $n = 3000$, the segments no. i and $i + 1$ intersect for $i = 1, \ldots, n - 1$;

`3ocen:` $n = 100\,000$, the interiors of all segments are disjoint (though one end-point may be shared);

`4ocen:` $n = 100\,000$, Byteasar can control all passenger in a single inspection.

In all sample grading tests $z = 1$.

Grading

The set of tests consists of the following subsets. Within each subset, there may be several unit tests. In the table below, z denotes the number of unit tests in a given subset and N stands for the sum of n over all unit tests in a subset.

If your program prints (for each unit test within a subset) the correct minimum number of ticket inspections, it will receive 20% of the score for the given subset. In such case, it should still print two numbers per unit test, where the second one should fit into 32 bit signed integer type.

Subset	**Property**	**Score**
1	$z \leq 10$, $n \leq 15$	10
2	$z \leq 100$, $N \leq 5000$	10
3	$z \leq 100$, $N \leq 500\,000$, all passengers can be controlled with at most three inspections	15
4	$z \leq 100$, $N \leq 500\,000$, the intersection of the interiors of any triplet of segments is empty	15
5	$z \leq 100$, $N \leq 500\,000$	50