1403 – Air Raid

Consider a town where all the streets are one-way and each street leads from one intersection to another. It is also known that starting from an intersection and walking through town's streets you can never reach the same intersection i.e. the town's streets form no cycles. With these assumptions your task is to write a program that finds the minimum number of paratroopers that can descend on the town and visit all the intersections of this town in such a way that no intersection is visited by more than one paratrooper. Each paratrooper lands at an intersection and can visit other intersections following the town streets. There are no restrictions on the starting intersection for each paratrooper.

Input

Input starts with an integer **T (≤ 100)**, denoting the number of test cases.

Each case starts with a blank line. Next line contains two integers **n (1 ≤ n ≤ 1000)** and **m (0 ≤ m ≤ 10000)** where **n** denotes the number of intersections and **m** denotes the number of one-way streets. Intersections are numbered from **1** to **n**. Each of the next **m** lines contains two integers **u v (1 ≤ u, v ≤ n, u ≠ v)** denoting a one-way street from intersection **u** to **v**. There can be at most one street from one intersection to another.

Output

For each case, print the case number and the minimum number of paratroopers required to visit all the intersections in the town.

Sample Input	Output for Sample Input
3 4 3 3 4 1 3 2 3 3 3 1 3 1 2 2 3 5 4 1 2 3 2 2 4 2 5	Case 1: 2 Case 2: 1 Case 3: 3

Note

Dataset is huge, use faster I/O methods.