Code Jam 2014 - Round 1B

The Bored Traveling Salesman

Problem

Your boss is sending you out on an international sales trip. What joy!

You have **N** cities (numbered from 1 to **N**) to visit and can get to them using a set of bidirectional flights that go between the cities.

All of the cities must be visited at least once. To do this you can book any number of tickets, subject to the following conditions:

- Each ticket consists of 2 flights, one from a specific city **X** to another specific city **Y** (this is called the **outbound** flight), and the other one from city **Y** to city **X** (this is called the **return** flight).
- You must use the outbound flight before the corresponding return flight (you can use other flights in between).
- At most 1 outbound flight going to each city, although there is no limit on the return flights (multiple return flights can go to the same city).
- You must use all flights which belong to the tickets you booked.
- You can otherwise visit the cities in any order you like.
- You can start your trip from any city you choose. You may not take an outbound flight to your starting city.

Now you could try to minimize the total distance travelled, but you did that last time, so that would be boring. Instead you noticed that each city has a distinct 5 digit ZIP (postal) code. When you visit a city for the first time (this includes the city which you start from) you write down the zip code and concatenate these into one large number (concatenate them in the order which you visited each city for the first time). What is the smallest number you can achieve?

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow.

Each test case starts with a single line containing two integers: the number of cities **N** and the number of possible bidirectional flights **M**.

N lines then follow, with the i-th line containing the 5-digit zip code of the i-th city. No ZIP code will have leading zeros and all ZIP codes in each test case will be distinct.

M lines then follow, each containing two integers **i** and **j** ($1 \leq i < j \leq N$) indicating that a bidirectional flight exists between the **i**-th city and the **j**-th city. All flights will be distinct within each test case.

It is guaranteed that you can visit every city following the rules above.

Output

For each test case, output one line containing "Case #x: y", where x is the test case number (starting from 1) and y is the smallest number you can achieve by concatenating the ZIP codes along your trip.

Limits

Memory limit: 1 GB.
1 ≤ **T** ≤ 100.
0 ≤ **M** ≤ **N** * (**N** - 1) / 2.

Small dataset

Time limit: 60 seconds.
1 ≤ **N** ≤ 8.

Large dataset

Time limit: 120 seconds.
1 ≤ **N** ≤ 50.

Sample

Sample Input

```
4
3 2
10001
20000
10000
1 2
2 3
5 4
36642
28444
50012
29651
10953
1 4
2 3
2 5
4 5
5 5
36642
28444
50012
29651
10953
1 2
1 4
2 3
2 5
4 5
6 6
10001
10002
10003
10004
10005
```

Sample Output

```
Case #1: 100002000010001
Case #2:
109532844450012296513664 2
Case #3:
109532844436642296515001 2
Case #4:
100011000210003100041000510006
```

```
10006
1 2
1 6
2 3
2 4
3 5
4 5
```

Explanation

In the last sample test case, the following is the sequence of what you should do to achieve the smallest number:

1. Start from city 1, write 10001.
2. Outbound flight from 1 to 2, write 10002.
3. Outbound flight from 2 to 3, write 10003.
4. Return flight from 3 to 2.
5. Outbound flight from 2 to 4, write 10004.
6. Outbound flight from 4 to 5, write 10005.
7. Return flight from 5 to 4.
8. Return flight from 4 to 2.
9. Return flight from 2 to 1.
10. Outbound flight from 1 to 6, write 10006.
11. Return flight from 6 to 1.