19I510 DESIGN AND ANALYSIS OF ALGORITHMS LABORATORY

LAB TEST - 1

SET - 1

1. Ramu is a night-guard at the "Lavish office buildings Ltd." headquarters. The office has a large grass field in front of the building. So every day, when Ramu comes to duty at the evening, it is his duty to turn on all the lights in the field. However, given the large size of the field and the large number of lights, it is very tiring for him to walk to each and every individual light to turn it on. So he has devised an ingenious plan — he will swap the switches for light-sensitive triggers. A local electronic store nearby sells these funny trigger switches at a very cheap price. Once installed at a light-post, it will automatically turn that light on whenever it can sense some other light lighting up nearby. So from now on, Ramu can just manually flip a few switches, and the light from those will trigger nearby sensors, which will in turn light up some more lights nearby, and so on, gradually lighting up the whole field. Now Ramu wonders: how many switches does he have to flip manually for this?

Input

The input starts with an integer T, the number of test cases to follow. Each test case will start with two integers N ($1 \le N \le 10000$) and M ($0 \le M \le 100000$), where N is the number of lights in the field, and M more lines of input follows in this input case. Each of these extra M lines will have two integers a and b separated by a space, where $1 \le a, b \le N$, indicating that if the light lights up, it will trigger the light b to turn on as well (according to their distance, brightness, sensor sensitivity, orientation and other complicated factors).

Output

For each input test case, the output must be a single line of the format 'Case k: c' where k is the case number starting with 1, and c is the minimum number of lights that Ronju must turn on manually before all the lights in the whole field gets lit up.

Sample Input

```
2
5 4
1 2
1 3
3 4
5 3
4 4
1 2
1 3
4 2
4 3
```

Sample Output

```
Case 1: 2
Case 2: 2
```

Input	Output
1 3 3 2 1 1 3 3 2	Case 1: 1
1 6 5 1 2 4 2 3 2 5 2 2 6	Case 1: 4

2. One measure of "unsortedness" in a sequence is the number of pairs of entries that are out of order with respect to each other. For instance, in the letter sequence "DAABEC", this measure is 5, since D is greater than four letters to its right and E is greater than one letter to its right. This measure is called the number of inversions in the sequence. The sequence "AACEDGG" has only one inversion (E and D) — it is nearly sorted — while the sequence "ZWQM" has 6 inversions (it is as unsorted as can be — exactly the reverse of sorted). You are responsible for cataloging a sequence of DNA strings (sequences containing only the four letters A, C, G, and T). However, you want to catalog them, not in alphabetical order, but rather in order of "sortedness", from "most sorted" to "least sorted". All the strings are of the same length.

   Input: The first line contains two integers: a positive integer n ($0 < n \leq 50$) giving the length of the strings; and a positive integer m ($0 < m \leq 100$) giving the number of strings. These are followed by m lines, each containing a string of length n.

   Output: Print the list of input strings, arranged from "most sorted" to "least sorted". If two or more strings are equally sorted, list them in the same order they are in the input file

   Sample Input:
   10 6
   AACATGAAGG
   TTTTGGCCAA
   TTTGGCCAAA
   GATCAGATTT
   CCCGGGGGGA
   ATCGATGCAT

   Sample Output:
   CCCGGGGGGA
   AACATGAAGG
   GATCAGATTT
   ATCGATGCAT
   TTTTGGCCAA
   TTTGGCCAAA

Input	Output
12 10 TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA	TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA TGCATGCATGCA
10 13 AACATGAAGG TTTTGGCCAA TTTGGCCAAA GGGGGGGGGG GATCAGATTT CCCGGGGGGA TTTTTTTTTT ATCGATGCAT AAAAAAAAAA CCCCCCCCCC ACGTACGTAC CGTACGTACG TGCATGCATG	GGGGGGGGGG TTTTTTTTTT AAAAAAAAAA CCCCCCCCCC CCCGGGGGGA AACATGAAGG GATCAGATTT ACGTACGTAC ATCGATGCAT CGTACGTACG TGCATGCATG TTTTGGCCAA TTTGGCCAAA
10 6 AACATGAAGG TTTTGGCCAA TTTGGCCAAA GATCAGATTT CCCGGGGGGA ATCGATGCAT	CCCGGGGGGA AACATGAAGG GATCAGATTT ATCGATGCAT TTTTGGCCAA TTTGGCCAAA
7 5 CTGAGCC GCCACGG TATGCGT ATCTCGT GGCTGTT	GGCTGTT ATCTCGT GCCACGG TATGCGT CTGAGCC

3. Write a program which takes text for an anonymous letter and text for magazine and determines if it is possible to write the anonymous letter using the magazine. The anonymous letter can be written using the magazine if for each character in the anonymous letter, the number of times it appears in the magazine.

   Use hashing to reduce the time complexity of the task

Input: The input is provided in two lines. The first line contains the text for anonymous letter and the second line contains the text for the message.

Output: Print YES if the letter can be written using the magazine (or) print NO along with the lexicographically smallest character which occurs more number of times in the anonymous letter than the magazine.

Sample Input
abcd
fghi
Sample Output
NO a

Input	Output
Hello Hardertosurvivewhenworldisnotloyal	YES
Hello Hardertosurvivewhenworld	NO l
Destiny sourcedestination	NO Y

4. Given a string and the pattern, determine the index positions where the pattern is found in the string. Provide O(n) solution for the problem

Input: The first line contains a string and the following line contains the pattern

Output: Print all the index positions where the pattern is found separated by a space. If the pattern is not found in the string, print -1.

Sample Input:
health is wealth
ealth

Sample Output:
1 11

Input	Output
Time and Tide Wait for None Time	0
ADADADDDDADAD ADAD	0 2 9
STUDIOUS STUDENT STUD	0 9
LOTOFMONEYINLOTTERY LOT	0 12