Greetings From Globussoft

- Given below are 5 Programming questions, you have to solve any 3 out of 5 questions.

- These 5 questions you can attempt in any technology like C/C++, java, .Net, PHP

- To solve these 3 questions you've max. 3 hours.

- While Solving these questions you are not allowed to use any Search Engine like Google, Yahoo, Bing ...

All the best for your test

Globussoft

QUESTION - 1

Some of the more elite (and not-so-elite) coders around take part in a certain unnamed programming contest. In said contest, there are multiple types of competitions. Here, we consider the Open and High School competition types. For each type, each competitor receives a *rating*, an integer between 1 and 100000, inclusive. A coder's rating is based upon his or her level of performance in matches and is calculated using a complicated formula which, thankfully, you will not be asked to implement.

Although the Open and High School ratings for a coder who has participated in both competition types lately are usually close, this is not always the case. In particular, High School matches are more about speed, since many coders are able to solve all the problems, whereas Open matches require more thinking and there is a steeper curve in terms of problem difficulty.

Problem Statement
You are given N coders ($1 \leq N \leq 300000$), conveniently numbered from 1 to N. Each of these coders participates in both High School and Open matches. For each coder, you are also given an Open rating A_i and a High School rating H_i. Coder i is said to be *better* than coder j if and only if both of coder i's ratings are greater than or equal to coder j's corresponding ratings, with at least one being greater. For each coder i, determine how many coders coder i is better than.

Input Format
On the first line of input is a single integer N, as described above.
N lines then follow. Line i+1 contains two space-separated integers, A_i and H_i.

Output Format
Line i should contain the number of coders that coder i is better than.

Sample Input

```
8
1798 1832
862 700
1075 1089
1568 1557
2575 1984
1033 950
1656 1649
1014 1473
```

Sample Output

```
6
0
2
4
```

```
7
1
5
1
```

QUESTION – 2

FJ has N ($1 \leq N \leq 50,000$) cows and M ($1 \leq M \leq 50,000$) bulls. Given a list of P ($1 \leq P \leq 150,000$) potential matches between a cow and a bull, compute the greatest number of pairs that can be matched. Of course, a cow can be matched to at most one bull, and vice versa.

Input

The first line contains three integers, N, M, and P. Each of the next P lines contains two integers A ($1 \leq A \leq N$) and B ($1 \leq B \leq M$), denoting that cow A can be matched with bull B.

Output

Print a single integer that is the maximum number of pairs that can be obtained.

Example

Input:

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

Output:

```
3
```

QUESTION – 3

Japan plans to welcome the ACM ICPC World Finals and a lot of roads must be built for the venue. Japan is tall island with N cities on the East coast and M cities on the West coast ($M \leq 1000$, $N \leq 1000$). K superhighways will be build. Cities on each coast are numbered 1, 2, ... from North to South. Each superhighway is straight line and connects city on the East coast with city of the West coast.

The funding for the construction is guaranteed by ACM. A major portion of the sum is determined by the number of crossings between superhighways. At most two superhighways cross at one location. Write a program that calculates the number of the crossings between superhighways.

The input file starts with T - the number of test cases. Each test case starts with three numbers – N, M, K. Each of the next K lines contains two numbers – the numbers of cities connected by the superhighway. The first one is the number of the city on the East coast and second one is the number of the city of the West coast.

For each test case write one line on the standard output:

Test case "case number": "number of crossings"

Sample

```
Input :
1
3 4 4
1 4
2 3
3 2
3 1
Ouput:
Test case 1: 5
```

QUESTION – 4

A chemical company has an unusual shortest path problem. There are N depots (vertices) where chemicals can be stored. There are M individual shipping methods (edges) connecting pairs of depots. Each individual shipping method has a cost. In the usual problem, the company would need to find a way to route a single shipment from the first depot (0) to the last (N - 1). That's easy. The problem they have seems harder. They have to ship two chemicals from the first depot (0) to the last (N - 1). The chemicals are dangerous and cannot safely be placed together. The regulations say the company cannot use the same shipping method for both chemicals. Further, the company cannot place the two chemicals in same depot (for any length of time) without special storage handling --- available only at the first and last depots.

To begin, they need to know if it's possible to ship both chemicals under these constraints. Next, they need to find the least cost of shipping both chemicals from first depot to the last depot. In brief, they need two completely separate paths (from the first depot to the last) where the overall cost of both is minimal. Your program must simply determine the minimum cost or, if it's not possible, conclusively state that the shipment cannot be made.

The input will consist of multiple cases. The first line of each input will contain N and M where N is the number of depots and M is the number of individual shipping methods. You may assume

that N is less than 20 and that M is less than 200. The next M lines will contain three values, i, j, and v. Each line corresponds a single, unique shipping method. The values i and j are the indices of two depots, and v is the cost of getting from i to j. Note that these shipping methods are directed. If something can be shipped from i to j with cost 10, that says nothing about shipping from j to i. Also, there may be more than one way to ship between any pair of depots, and that may be important here.

A line containing two zeroes signals the end of data and should not be processed.

Sample

```
Input :
2 1
0 1 20
2 3
0 1 20
0 1 20
1 0 10
4 6
0 1 22
1 3 11
0 2 14
2 3 26
0 3 43
0 3 58
0 0
Ouput:
Instance #1:  Not possible
Instance #2:  40
Instance #3:  73
```

QUESTION – 5

There is a checkmates board with **n** rows and **m** columns. Some of the cells of the board are occupied. There is a queen standing on a certain cell. It wants to move to another cell of this board. Help it do this making the least possible moves. The queen can go any number of cells in any of eight directions in a single move, but it can't pass through or stand on the occupied cells and leave the board.

Input

The first line of the input contains number **t** – the amount of tests. Then **t** test descriptions follow. The first line of each test consists of two numbers **n** and **m** separated with a space. Then **n** lines follow each containing **m** characters describing the board. Character '.' means a free cell, character 'X' – an occupied cell, character 'S' – the starting cell of the queen, character 'F' – the

cell where the queen wants to go. It is guaranteed that there will be exactly one character ‘S’ and one character ‘F’ on each board.

Constraints

1 <= **t** <= 30
2 <= **n**, **m** <= 1000

Output

For each test case print the minimum number of moves the queen has to do to reach the desired cell. Print ‘-1’ if the queen can’t reach the cell.

Example

Input:

```
3
3 3
S..
...
..F
3 3
S..
XX.
F..
3 3
S..
XXX
..F
```

Output:

```
1
3
-1
```