Problem A. Alice and Maze

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

Alice likes to collect things. She likes all kinds of things including constest problems, balloons, other teams' mascots and ACs. One day she was collecting balloons on the grand line. She noticed that seemingly she got lost in a labyrinth.

Foutunately, one mascot she got earlier is a map of the labyrinth. After some research, she found out where she was and where was the exit. She also noticed there were still some pieces of treasure in the labyrinth waiting for her, and they seemed to respawn indefinitely each time she got to a place with a treasure. However, the tunnels between those treasures were occupied by mischievous rabbits who would charge Alice a part of her belongings each time she would traverse a tunnel. If Alice doesn't have enough treasure to pay the toll than she's not able to use the tunnel at that moment.

Alice needs you to help her find a way to the exit with as much treasure as possible. Although she already has some treasure, she still wants to maximize the amount of treasure she has when she gets to the exit. If Alice may collect an arbitrarily large amount of treasure on her way to the exit, output -1. If there is no way to get to the exit, output -2 (recall that Alice has to have enough treasure to pay the toll for each tunnel she passes).

Input

The first line contains an integer T indicating the total number of test cases. Each test case begins with three integers n, m, q, denoting the number of treasure spots, the number of tunnels, and the amount of treasure Alice has initially. The following line contains n integers g_i, denoting the amount of treasure Alice can acquire each time she reach the treasure spot i. Each of the following m lines contains three integers u_j, v_j, p_j, indicating that there is a tunnel between the treasure spots u_j and v_j, and Alice needs to pay p_j to pass through that tunnel in either direction. Initially Alice is located at the treasure spot 1. The exit is located at the treasure spot n.

You may assume that:

- $T \leq 10$
- $2 \leq n \leq 1000$
- $2 \leq m \leq 10^5$
- $1 \leq u_i, v_i \leq n$
- $1 \leq q, g_i, p_j \leq 10^7$
- There won't be multiple tunnels between any pair of treasure spots, nor a tunnel connecting a treasure spot to itself.

Output

For each test case, output:

- -2 if there is no way to reach the exit;
- otherwise, -1 if Alice can reach the exit with an arbitrarily large amount of treasure;
- otherwise, the largest possible amount of treasure Alice can have upon reaching the exit.

Example

standard input	standard output
3 2 1 3 4 9 2 1 8 3 3 5 2 9 4 1 2 6 1 3 5 2 3 7 3 2 0 1 2 0 1 2 1 2 3 100	-2 7 -1

Problem B. Buggy Stack Machine

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	512 mebibytes

A stack machine uses a stack to store operands, and all operators are applied on the topmost operands. For example, addition is a binary operator, and it will pop two operands from the stack and put their sum back to the stack. Thor Cooperation manufactures a type of stack machine which only accepts 32-bit signed integers as operands and has addition as its only operator. We use 2's complements to represent negative numbers, and when addition causes an overflow, we just drop the carry bit. Its machine language is quite simple. There are only four instructions as described below.

1. `PUSH` x: push a 32-bit singed integer x into the stack.
2. `POP`: remove the operand on the top of the stack, then print it out on a line. If the stack is empty, the stack machine outputs `ERROR` and halts.
3. `ADD`: Remove top two operands and put their sum back to the stack. If the stack has less than two operands, the stack machine output `ERROR` and halts.
4. `EMPTY`: Clear the stack.

However, the machine has a defect. If the topmost k operands have smallest bit values $b_1, \ldots, b_k$ (b_i is the value of the i-th topmost operand's smallest bit), then all these k operands will be wiped out immediately. Given $k, b_1, \ldots, b_k$ and a sequence of instructions, please write a program to simulate the buggy stack machine which has an empty stack initially.

Input

The first line of the input contains a positive integer T indicating the number of test cases. Each test case consists of many lines. The first of a test case contains two positive integers k and n. The second line contains k integers $b_1, \ldots, b_k$ which are the smallest bit values triggering the erasure of operands. The following n lines are the sequence of instructions, and there is only one instruction per line.

You may assume:

- $1 \leq T \leq 50$
- $n \leq 10^5$
- $k \leq 10^4$
- $b_1, \ldots, b_k \in \{0, 1\}$
- An instruction can only be in the following format: `PUSH` x, `POP`, `ADD`, `EMPTY`.
- The size of input is less than 60 mebibytes.

Output

For the `i`-th test case, you should output a line "`CASE i:`" before the output of the buggy stack machine. If the machine encounters an error, then it should output "`ERROR`" and ignore all following instructions in the test case.

Example

standard input	standard output
3 2 5 0 0 PUSH 3 PUSH 5 ADD POP POP 2 3 0 0 PUSH 2 PUSH 4 ADD 2 5 0 0 PUSH 2 EMPTY PUSH 4 PUSH 5 POP	CASE 1: 8 ERROR CASE 2: ERROR CASE 3: 5

Problem C. Constellation

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Recently explorers discovered an ancient scroll which describes some constellation. It suggests that the constellation consists of M stars and contains the distances between each pair of those stars. The distances are measured as if the sky was a plane and the stars were points on it.

There are N stars visible in that hemisphere of the present-day sky where the constellation is supposed to be. Usually, a constellation comprises of the brightest stars of some fragment on the sky, but the brightnesses of the stars could have changed over the past milleniums, so this won't help. Therefore, identifying the stars on present-day sky which were in the constellation should be performed on the basis of the given distances only.

A possible location of the constellation is a list of stars $(I_1, I_2, ldots, I_M)$ such that for each i and j $(1 \le i, j \le M)$ the distance between stars I_i and I_j equals the distance between the i-th and j-th stars of the scroll. Two locations are different if there is at least a single position in the lists where stars differ.

You're given a set of stars on the present-day sky. Also, you're given an $M \times M$ matrix, where element (i, j) is the squared distance between star i and star j of the constellation. Count the number of different possible locations of the given constellation.

Input

The first line of the input contains a single number M. The next M lines contain M numbes each — the distance matrix (the given distance matrix is symmetric, its main diagonal contains zeroes only, and the rest of numbers are positive and do not exceed 10^8).

Next line contains number N. Each of the next N lines contains a couple of numbers X_i, Y_i — the coordinates of the i-th star on present-day sky ($2 \le N \le 3 \cdot 10^4$, $2 \le M \le min(N, 20)$). The coordinates of each star are integers not exceeding 10^4 by absolute value. No two stars are in the same point.

Output

Print the number of possible locations of the constellation from the scroll.

Example

standard input	standard output
3 0 1 2 1 0 1 2 1 0 4 0 0 1 0 0 1 1 1	8

Note

The possible locations of the constellation are (1,2,4), (1,3,4), (2,1,3), (2,4,3), (3,1,2), (3,4,2), (4,2,1), (4,3,1).

Problem D. Domination

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

The Nowhereland is a country with an advanced parliamentary system. As with all parliaments, there are no moral, no standards and no justice. There is but one thing, the pursuit of power. In politics, almost everyone has dirt on someone. If you have destructive information on someone else, you dominate that person. Once you **dominate** someone, you can get them to do whatever you want. This is how the game of politic is played. So naturally, if there exists a group of Members of the Parliament (also known as MPs), in which everyone in the group dominates everyone else outside the group, this group clearly rules the entire parliament. Such a group is called a **dominant group**. They are the only ones that really matter, they alone will decide everything. Among them, the smallest dominant group is called the **Top Cycle**, now this is the true center of power, the ultimate goal of all politicians. Given the relation of the MPs of the Nowhereland, please find out the true leader(s) of this nation.

Note that it is guaranteed that for each pair of MPs x and y either x dominates y or y dominates x.

Input

On the first line there is a single integer T ($T \leq 15$) indicating the number of test cases. The first line of each test case contains one integers n ($0 < n \leq 1000$) indicating the size of the parliament. The following n line contains n integers, each of them are either -1, 1 or 0, as a n by n matrix denoting the relationship between MPs. If the number on the i-th row and the j-th column is 1, that means MP i dominates MP j, and if the number is -1, then MP j dominates MP i. 0 appears only on the diagonals, and the diagonals must be 0. It is guaranteed that one MP either dominates or is dominated by another MP. That is, two MPs cannot dominate (or be dominated by) each other at the same time.

Output

For each test case, output the size of the Top Cycle. There should be a line break at the end of the output of each test case.

Example

standard input	standard output
2 3 0 1 1 -1 0 1 -1 -1 0 3 0 1 -1 -1 0 1 1 -1 0	1 3

Problem E. Elevator

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

The tallest building in our university has 202 floors and a smart elevator. In order to save energy, the smart elevator will stop at as less floors as possible. It observes that all visitors are willing to walk from floor i to floor $i-1$ for $i \in \{2, 3, \ldots, 202\}$, and it also finds that no visitors are willing to walk from floor i to floor j if $j \notin \{i-1, i\}$. Therefore, it can just stop at floor i if there are visitors going to floor i and visitors going to floor $i-1$.

Now, there are n visitors just arrive floor 1, and the destination of visitor j is floor a_j. Except floor 1, how many floors should the smart elevator stop at?

Input

The first line of the input contains a positive integer T indicating the number of test cases. Each test case consists of two lines. The first line is a line containing exactly one non-negative integer n where n is the number of visitors just arriving floor 1. The second line contains n integers $a_1, \ldots, a_n$ separated by blanks.

You may assume:

- $1 \leq T \leq 100$
- $0 \leq n \leq 10^4$
- $a_1, \ldots, a_n \in \{1, 2, \ldots, 202\}$

Output

For each test case, output the number of floors at which the smart elevator stop except floor 1.

Example

standard input	standard output
2 5 1 2 3 4 4 10 1 2 4 5 7 9 10 10 1 1	2 4

Problem F. Flow Strikes Back

Input file:	*standard input*
Output file:	*standard output*
Time limit:	7 seconds
Memory limit:	512 mebibytes

Let $G = (V, E)$ be a directed network defined by a set V of n nodes (vertices) and a set E of m directed arcs (edges). We associate an integer number b_i with each node $i \in V$. You have to solve the following problem given in terms of optimization model:

Minimize $z(x) = \max\limits_{(i,j) \in E} x_{ij}$,

subject to

$$\sum_{\{j:(i,j)\in E\}} x_{ij} - \sum_{\{j:(j,i)\in E\}} x_{ji} = b_i \qquad \text{for all } i \in V,$$

$$x_{ij} \text{ is nonnegative integer for all } (i,j) \in E.$$

Network G doesn't contain loops (it means that $(i,i) \notin E$ for all $i \in V$) and between any two vertices $i \in V$ and $j \in V$ there can be at most one directed arc $(i,j) \in E$ and at most one directed arc $(j,i) \in E$.

Input

On the first line of the input file there are two integers n and m. Assume that $2 \le n \le 10\,000$ and $1 \le m \le 500\,000$.

The second line contains the sequence of integers b_i $(i = 1, 2, \ldots, n)$ separated by single spaces. $|b_i| \le 10^6$ for all $i = 1, 2, \ldots, n$.

The following m lines give information about graph structure: each line contains two numbers i and j separated by a single space. These numbers mean that the directed arc (i,j) is in E.

Output

Output the optimal value of $z(x)$. It is guaranteed that a solution exists.

Examples

standard input	standard output
4 5 10 0 0 -10 1 2 2 4 1 3 3 4 1 4	4
4 3 10 -1 -7 -2 1 2 1 3 3 4	9

Problem G. Good Substring

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Let's call a string *good*, if none of its substings of length more than 1 is a palindrome.

You are given a string S. After each of Q requests to change a symbol in the string, print the length of a longest good substring of S.

Input

First line of the input contains two integers N and Q ($1 \leq N, Q \leq 10^5$) — length of the string S and number of queries Q. Second line contains string S of length N, consisting of lowercase English letters. Then Q queries follow. Each of next Q lines contains integer i ($1 \leq i \leq N$) and lowercase English letter c, describing that at this query i-th character in the string is replaced with character c.

Output

For each query print one integer — the length of longest good substring after the replacement.

Example

standard input	standard output
5 3 abcde 5 f 3 a 4 b	5 4 3

Problem H. House Selection

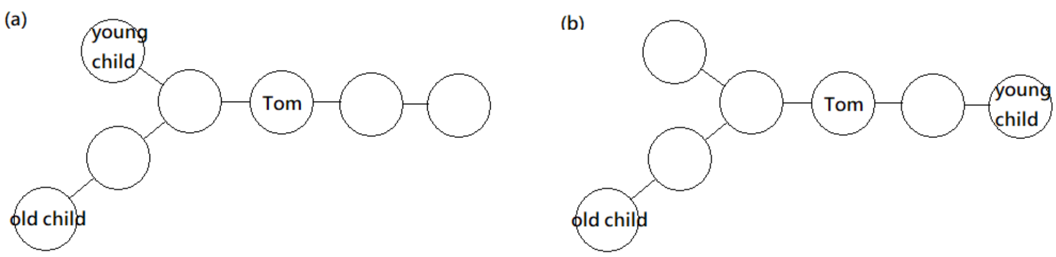
Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

Tom lives in X community. The structure of X community is a tree graph, with n nodes and $n-1$ roads. Each house is located on a node, and the nodes are connected by roads. Every node has exactly one house and every road is exactly 1 km, connecting different two nodes to make the whole community a "tree" structure.

Every year, the residents of X community will change their locations with each other by randomly drawing lots. That is, everyone in X community will randomly receive a new number corresponding to the house they will live in in the coming year. The numbers assigned to everyone are different.

Tom has two children. Both of his children live in X community, too. Tom thinks that it will be wonderful if the new arrangement meets all the conditions below:

1. His older child lives exactly p km away from him.
2. His younger child lives exactly q km away from him.
3. The path from his older child to him and the path from his younger child to him don't overlap, that is, his children will never meet on the road if they want to visit their father at the same time.



Look at the examples above, while $p = 3$ and $q = 2$, both example (a) and example (b) meet the first two conditions. However, Tom's children may meet each other in example (a). Thus, only example (b) meets Tom's favor.

However, the arrangement of their new residential location cannot be decided by anyone, but by the result of randomly drawing lots. Can you tell Tom what is the probability that the arrangement meets his favor?

Input

The first line of input contains a single integer T indicating the number of test cases.

Each test case starts with a line with 3 integers, n, p, q, indicating the number of nodes in X community, and the desired distances to his children.

Each of the following $n-1$ lines contains two integers a_i and b_i indicating that there is a road connecting node a_i and node b_i.

You may assume that:

- $1 \le T \le 20$
- $3 \le n \le 50000$
- $0 < p \le 50, 0 < q \le 50$

- For $i \in \{1, 2, \ldots, n-1\}$, $1 \le a_i \le n$ and $1 \le b_i \le n$.

Output

For each test case, output one line with the probability that the random arrangement meets Tom's favor. You may output numbers in scientific notation. For example, `1.234e-5` is equivalent to `0.00001234`. Your answer will be rejected if the both absolute and relative errors are greater than 10^{-7}.

Example

standard input	standard output
2 5 1 1 1 2 2 3 3 4 2 5 5 2 1 1 2 2 3 2 4 4 5	0.13333333333333333 0.06666666666666667

Problem I. Intelligence and Magic

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

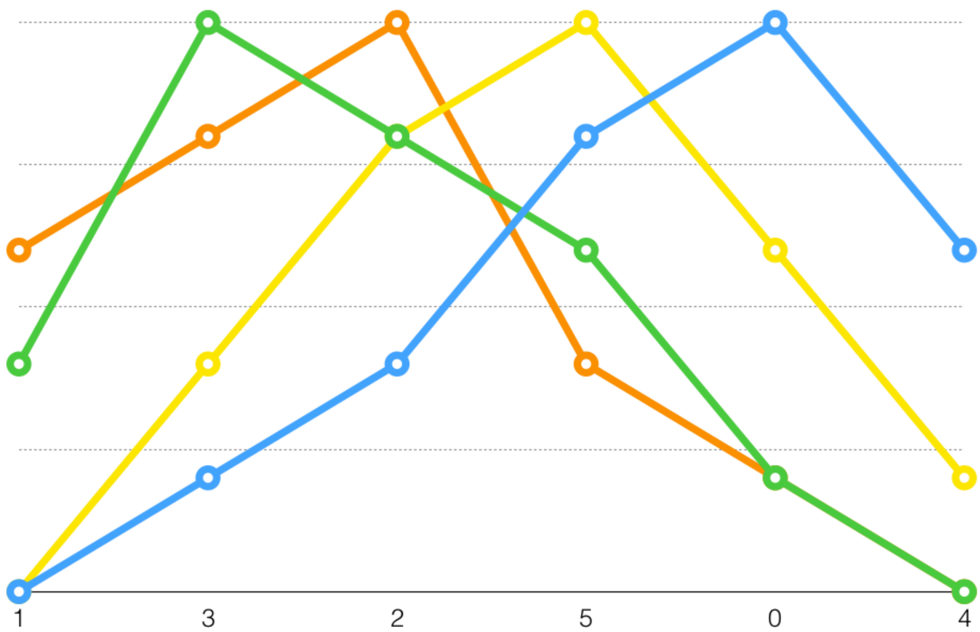
You are an undercover secret agent of a up and coming new magic society. Being a spy sounds cool, but nowadays spies don't go on and fight enemies or hook up with someone wherever they go. Those spies exist only in the movies. (Yes, even a agent from magic society watches Hollywood.) No, being a spy is about being a normal person doing normal stuff, and gather information that seems harmless enough. But looking at them from an eye of an expert, even the most useless piece of clue can worth their weight in gold.

Now, you are pretending to be a harmless, boring librarian who takes care of the ancient city library which don't even have Wi-Fi. Obviously, no muggle ever come here anymore. Especially after the new library is built. However, don't be fooled by its ordinary look. Back in the old days when magic and still entangled in the daily life of common muggles, a lot of secrets are carelessly left here. Here you find an old documents about auction records five centuries ago where a lot of ancient magic societies were bidding for lands to establish their base.

As a member of a magic society you clearly know that every place have a different amount of holy and shadow energy known as their attributes. Different magic societies prefer different balance of the two energy. However, in modern times, the attribute of each location became a heavily guarded secret. If you can figure out the attribute of these locations, your magic society will benefit greatly.

Assume that we can order all these potential locations in a linear order, such that its attribute changes from holy to shadow linearly. Since we know that each magic society prefers a specific balance of energy the most, so we may assume that for each society, their preference will go up from the one end of the attribute spectrum until it reaches their optimal location, and then go down to the other end.

Formally, given a linear order of location x, y, z where $x < y < z$ or $z < y < x$. If x is prefered over y, then y must also be prefered over z. Now from the records you managed to compile a list of preference order of locations for each magical society. Can you also find out the possible linear ordering for the locations that fit these preference order? For instance, following is the solution for sample input 2, where the ordering $(1, 3, 2, 5, 0, 4)$ achieves "single-peakness" for every single preference.



Input

The input starts with an integer T, where T represents the number of test cases. Each test case will start with two integers N and M, representing the number of magic societies and the number of locations respectively. It is followed by N lines each representing the preference of a magic society, containing M integers $x_1, x_2, \ldots, x_M$ indicating that this magic society prefers x_1 over x_2 over x_3 and so on.

You may assume:

- $1 \leq T \leq 20$
- $0 < N \leq 20$ and $0 < M \leq 100$

Output

Output M integers on one line representing the linear ordering of the locations. If there exists more than one possibilities, output the one that is the least lexicographically. If there doesn't exist any possible linear ordering, then you must have mixed up some muggle companies in the list, in that case just output "Muggles!" on the line.

Example

standard input	standard output
3 3 6 0 1 2 5 3 4 0 3 1 2 5 4 1 3 0 4 2 5 4 6 0 5 4 2 3 1 3 2 5 1 0 4 5 2 0 3 4 1 2 3 1 5 0 4 4 4 0 1 2 3 0 2 1 3 1 3 2 0 0 1 3 2	Muggles! 1 3 2 5 0 4 Muggles!

Problem J. Jack Barmer

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Jack Barmer is a special agent of the Counter Terrorist Unit (CTU). He got intel about terrorists infiltrating a highly classified cutting-edge biotechnology lab. He went in and shoot down all the terrorists single handedly, however he was too late. The terrorists have already released the deadly nerve gas. Fortunately, the quarantine mechanism is still intact. But it won't last long. The nerve gas is corrosive! The lab is a rectangular building made up of n by m cells and each cell is connected to its neighboring cells. The CTU has already pulled out the floor plan of the lab so Jack know exactly where the nerve gas is. Jack must put on protective clothes and enter the lab, activate the self-destruct system of each cell that is contaminated one by one to destroy the nerve gas before they are spread into the city. Note that whenever Jack enters a contaminated cell he must activate the self-destruct system, and after the system is activated, the cell is utterly destroyed...there is no way to re-enter that cell again. There is only one entrance and one exit in this lab. Time is ticking, what should Jack do?

Input

On the first line there is a single integer T indicating the number of test cases. The first line of each test case contains two integers n and m indicating the size of the lab. The following n line contains m characters, each of them are either '+' or 'X', representing that this is a normal or contaminated cell, respectively. The entrance is always at the top left most cell (0, 0), and the exit is always at bottom right most cell $(n-1, m-1)$.

You may assume:

- $1 \leq T \leq 20$
- $0 < n \leq 8$ and $0 < m \leq 8$

Output

For each test case, output an integer representing the least number of rooms jack needs to visit if it is possible to self-destruct all the contaminated cells and exit safely or "`impossible`" if not. Please note that you should count a room only once even if Jack visits it multiple times. Also, there should be a line break at the end of the output of each test case.

Example

standard input	standard output
2 4 4 XXXX XXXX XXXX XXXX 4 4 +XXX XXXX XXXX XXXX	impossible 16

Problem K. Knockout Tournament

Input file:	*standard input*
Output file:	*standard output*
Time limit:	15 seconds
Memory limit:	512 mebibytes

National Association of Tennis is planning to hold a tennis competition among professional players. The competition is going to be a knockout tournament, and you are assigned the task to make the arrangement of players in the tournament.

You are given the detailed report about all participants of the competition. The report contains the results of recent matches between all pairs of the participants. Examining the data, you've noticed that it is only up to the opponent whether one player wins or not.

Since one of your special friends are attending the competition, you want him to get the best prize. So you want to know the possibility where he wins the gold medal. However it is not so easy to figure out because there are many participants. You have decided to write a program which calculates the number of possible arrangements of tournament in which your friend wins the gold medal.

In order to make your trick hidden from everyone, you need to avoid making a factitive tournament tree. So you have to minimize the height of your tournament tree.

Input

The input consists of multiple datasets.

Each dataset has the format as described below.

```
N M
R11 R12 . . . R1N
R21 R22 . . . R2N
...
RN1 RN2 . . . RNN
```

N ($2 \leq N \leq 16$) is the number of player, and M ($1 \leq M \leq N$) is your friend's ID (numbered from 1). R_{ij} is the result of a match between the i-th player and the j-th player. When i-th player always wins, $R_{ij} = 1$. Otherwise, $R_{ij} = 0$. It is guaranteed that the matrix is consistent: for all $i \neq j$, $R_{ij} = 0$ if and only if $R_{ji} = 1$. The diagonal elements R_{ii} are just given for convenience and are always 0.

The end of input is indicated by a line containing two zeros. This line is not a part of any datasets and should not be processed.

Output

For each dataset, your program should output in a line the number of possible tournaments in which your friend wins the first prize.

Examples

standard input	standard output
2 1 0 1 0 0 2 1 0 0 1 0 3 3 0 1 1 0 0 1 0 0 0 3 3 0 1 0 0 0 0 1 1 0 3 1 0 1 0 0 0 0 1 1 0 3 3 0 1 0 0 0 1 1 0 0 6 4 0 0 0 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 1 0 0 1 0 1 0 0 0 0 0 0 1 1 1 1 0 7 2 0 1 0 0 0 1 0 0 0 1 0 1 1 1 1 0 0 1 1 0 0 1 1 0 0 0 1 0 1 0 0 1 0 0 1 0 0 1 0 1 0 0 1 0 1 1 0 1 0 8 6 0 0 0 0 1 0 0 0 1 0 1 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 1 0 0 1 0 1 0 1 0 1 0 0 1 0 1 1 1 0 1 0 0 1 1 1 1 1 0 1 0 0 1 1 1 0 1 0 1 0 0 0	1 0 0 3 0 1 11 139 78

Problem L. Love Power Plant

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Problem Description

A love power plant can generate a lot of energy. It is the cleanest power generator ever invented in the history. A love fuel stick is a string consisting of many `L`'s, `O`'s, `V`'s, and `E`'s. Love power plants generate power by consuming love fuel sticks as follows.

1. Find a subsequence `(L,O,V,E)` of the fuel stick, then remove it from the fuel stick to generate one gigawatt power.
2. If the fuel stick still has a subsequence `(L,O,V,E)`, then go to the previous step.

For example, `LLEOVEOLOVE` can generate at most two gigawatts power by the following steps.

1. The power plant remove the last four characters. This changes the fuel stick to `LLEOVEO`.
2. The power plant remove the 2nd, 4th, 5th, 6th characters. This change the fuel stick to `LEO`.

The government hires the best people to operate love power plants, so all fuel stick can generate as much energy as possible. Please write a program to determine how many gigawatts can be generated by a given fuel stick.

Input

The first line of the input contains a positive integer T indicating the number of test cases. Each test case is a line containing a love fuel stick.

You may assume:

- $1 \leq T \leq 100$
- All fuel stick are no longer than 100000.

Output

For each test case, output an integer P where the coressponding fuel stick can generate P gigawatts power.

Example

standard input	standard output
2 LOVE LLEOVEOLOVE	1 2