Problem A. Exam preperation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Students of one well-known university are planning to start their preparation for an exam in discrete mathematics. To prepare for the exam students need to solve m problems, using n theorems they have learned this semester. Theorems are numbered with integers from 1 to n and problems are numbered with integers from 1 to m. To solve i-th problem you need to apply theorems l_i,l_{i+1},$\dots$,r_i in that exact order. However, there is a catch. It is hard to remember theorems you are not using regularly. Each student has a value s that represents how good their memory is. If a student wants to apply a theorem that is not among s last theorems they had applied — they need to review their notes. Note, there may be duplicates among the s last theorems a student applied. Your task is to calculate how many times will each student need to review their notes in the process of solving m problems(problems should be solved in order 1,2,.....$n-1$,n).

Input

The first line of input contains three integers N,M,$Q$$(1 \leq N, M, Q \leq 2 \cdot 10^5)$ the number of theorems, problems and students respectively.

The following M lines contain two integers each l_i,$r_i$$(1 \leq l_i \leq r_i \leq N)$.

The last line contains Q integers s_1,s_2,$\dots$,$s_Q$$(0 \leq s_i \leq 10^{12})$.

Output

Output one line with Q space-seperated integers, the i-th integer represents the number of times i-th student will need to review his notes.

Example

standard input	standard output

`10 10 10`
`1 5`
`3 9`
`2 4`
`3 8`
`4 5`
`7 9`
`3 6`
`1 7`
`2 5`
`3 9`
`1 2 3 4 5 6 7 8 9 10`
	`39 37 31 29 27 17 17 15 13 9`

Note

Theorems that should be applied in this exact order:

1,2,3,4,5,3,4,5,6,7,8,9,

2,3,4,3,4,5,6,7,8,4,5,7,8,9,

3,4,5,6,1,2,3,4,5,6,7,2,3,4,5,

3,4,5,6,7,8,9

Problem B. Buy them all

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

Today you decided to buy some drones. Since the shop where you are planning to purchase them knows the reason why you need so many, they decided to give you M coupons. There are N drones in the shop, they are numbered from 1 to N. i-th drone normally costs A_i Ukrainian dollars. However, if you use one coupon on it, the price will drop to B_i. You can not use one coupon twice, and you can not use two coupons on one drone. So you can choose up to M drones and use coupons on them. You have S Ukrainian dollars and plan to buy as many drones as possible. Determine the maximum number of drones you can buy if you use your coupons and money optimally.

Input

The first line of the input contains three integers N,M and $S (1 \le N \le 50000, 0 \le M \le N, 1 \le S \le 10^{14})$.

The next N lines contain descriptions of drones, A_i,$B_i (1 \le A_i \le 10^9, 1 \le B_i \le A_i)$.

Output

Output one integer — the maximum number of drones you can buy.

Examples

standard input	standard output

`4 1 7`
`3 2`
`4 4`
`2 2`
`8 1`
	`3`

`7 2 10`
`4 3`
`3 3`
`4 1`
`6 3`
`8 4`
`5 5`
`6 2`
	`4`

Note

You can use your only coupon on a drone 4 and buy drones 1,3,4, spending $3 + 2 + 1 = 6$ Ukrainian dollars.

Problem C. Make it complete

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

The country of Byteland consists of N cities. There is a single bidirectional road between every two cities. Currently, M roads are open. All other roads are closed. The president of Byteland asked you to make the Byteland map complete, by doing the following operation any number of times: chooses one of the cities, open roads from that city towards all other cities where these roads do not currently exist, and at the same time close all existing roads from that city. Is it possible to open all $\frac{N(N-1)}{2}$ roads?

Input

The first line of input contains one integer $N(1 \le N \le 1000)$ — the number of cities.

The second line of input contains one integer $M(0 \le M < \frac{N(N-1)}{2})$ — the number of roads that are currently open.

The following m lines contain two integers each, x and y(1$\le$ $x, y \le N$, $x \ne y$) indicating that the road between cities x and y is currently open. Every road can appear on this list at most once.

Output

Output "YES"(without quotes) if it possible to open all roads in Byteland and "NO"(without quotes) otherwise.

Examples

standard input	standard output
2 0	YES

3
1
2 3
	YES

4
2
1 4
3 4
	NO

Problem D. Integer points

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given N line segments on the plane. Your task is to calculate the total number of points with integer coordinates that lie on at least one of the line segments.

Input

The first line of input contains one integer $N (1 \le N \le 5000)$ — the number of segments. Then there are N lines that contain a description of a segment: four integers x_1, y_1, x_2, $y_2 (-500 \le x_1, y_1, x_2, y_2 \le 500)$. Those four integers describe a segment with endpoints (x_1, y_1) and (x_2, y_2).

Output

Output one number — the answer to the problem.

Example

standard input	standard output

3
2 1 2 5
1 1 4 4
5 2 1 4
	10

Note

Those are 10 points with integer coordinates that lie on at least one segment: $(2, 1)$, $(2, 2)$, $(2, 3)$, $(2, 4)$, $(2, 5)$, $(1, 1)$, $(3, 3)$, $(4, 4)$, $(1, 4)$, $(5, 2)$.

Problem E. Generator

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

We're here for the explorers and the thrill seekers, for people looking for a little peace and quiet and the ones who can't wait to discover something new...

— Generator, *Official website*

The famous hotel (in fact hostel) **"Generator"** is hosting the competition with a tax, where the prize for the successful solution is 20% off the price for a one-night stay (one night because you for sure won't like to stay in that hotel for the second night).

Considering the fact that the hostel name is **"Generator,"** the task is about **generating**. Formally, for an integer n we consider subsets of set $1, 2, ..., n$. Given starting family H of subsets, we can **generate** new subsets with the following operations:

- choose two subsets A, B from H and add $A \cap B$ to H;
- choose two subsets A, B from H and add $A \cup B$ to H.

The **"Generator's"** goal is to generate all possible subsets of $1, 2, ..., n$. And since **"Generator"** would definitely like to save, your goal is to determine starting family H of a minimum size, which allows **generating** all possible subsets by performing the operations mentioned above.

Input

The first line contains a single number z, a number of test cases.

Each test case contains exactly one integer $n(2 \le n \le 200000)$. The sum of n over all test cases does not exceed 600000.

Output

For each test case, output an integer $|H|$, the starting family H size. Next $|H|$ lines should contain a description of the ith set: first p_i — that is the size of ith set. Then there should be p_i distinct integers from 1 to n — elements of the ith set, in any order.

Example

standard input	standard output

1
3
	3
	1 1
	1 2
	1 3

Problem F. Lawnmower

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	256 megabytes

Don Pablo has purchased new land and recently discovered there a rectangular $n \times m$ grass lawn divided into $n \cdot m$ unit squares. The unit square at the intersection of the i-th row and the j-th column contains a lawn of height $h_{i,j}$. Delighted with his find, Pablo decided to make the most of his good fortune. He went to the lawnmower shop, where there were s lawnmowers, k-th lawnmower is characterized with an integer p_k — height of it's blade.

It turns out that grass mowing is a very serious process with its own details. When a lawnmower with a height of the blade p_k is brought into a square containing grass of height $h_{i,j}$, then it behaves as follows:

- if $p_k \leq h_{i,j}$, nothing happens because the grass is too low for this lawnmower.
- if $p_k > 2 \cdot h_{i,j}$ lawnmower gets stuck and becomes broken
- otherwise, the lawnmower mows $h_{i,j} - p_k$ units of grass, lowering it's height to p_k

At every moment, Pablo, with his lawnmower, can be either outside of the lawn or in one of the unit squares. Pablo can move between the outside of the lawn and any of the squares at its edge (i.e., in the row or column that is either first or last); and from any square to any of its immediate four neighbors. Pablo would like to keep his lawnmower unbroken under all circumstances and temptations, and, at the same moment, he will mow as much grass as he can for obvious reasons.

Pablo doesn't have enough money (yet), so he will only buy one lawnmower. As he ponders his future prospects, he wants to know for each mower how much grass he can get. Help Paul as he comes up with the next use for the grass!

Input

The first line of input contains the number of test cases $z (1 \leq z \leq 2\ 000)$. The descriptions of the test cases follow. The first line of a test case contains two integers $n, m (1 \leq n \cdot m \leq 10^6)$ – the dimensions of the lawn. The next n lines contain m integers each – the grass heights in all $n \cdot m$ unit squares. The next line contains an integer $s (1 \leq s \leq 10^6)$ – the number of lawnmowers at the shop. The last line contains s integers – the blade heights of the lawnmowers. The heights of grass and all lawnmowers are between 1 and 10^9 inclusive. The total number of unit squares in all test cases, as well as the total number of lawnmowers, do not exceed $5 \cdot 10^6$ each.

Output

For each test case, output a single line containing s integers: the total amount of grass Don can mow for each lawnmower.

Example

standard input	standard output

1
3 4
1 9 6 2
8 5 9 9
9 8 7 4
5
2 1 8 4 3
	2 1 4 14 4

Note

All the unit squares in Professor's lawn that lie on the boundary are directly accessible from outside, so they can be mowed as long as the lawnmower has sufficient height. For example,$h_{1,2} = 9$ is too high for the 4-th lawnmower with $l_4 = 4$, but can be mowed with the 3-rd lawnmower that has $l_3 = 8$, yielding 1 unit of grass. Note that while $h_{2,2} = 5$ could technically be mowed with the 5-th lawnmower, this lawnmower has no way of accessing that square, as all of its direct neighbors are too high. With the 4-th lawnmower however, Professor can enter $(2, 2)$ from $(3, 2)$.

Problem G. Average problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

You are a typical average student. And as the average student's honor code instructs, you must choose to be average in everything, including length and size.

Today you choose the length of the pufix. The *pufix* of string t is its prefix, which does not equal the whole string t and is also its suffix (i.e., appears both at the beginning and at the end of t).

As you can already guess, you have no choice but to choose an average pufix. The *average pufix* of a string t (assuming t has at least one pufix) is the pufix whose length is the median of lengths of all pufixes of t.

The *median* of a sequence of numbers is the value appearing at the middle position once the sequence is sorted. In case sequence has even number of elements, the smaller number is selected. For example, the median of the sequence $[2, 1, 10, 6, 9, 100]$ is 6.

Now, do your average duty! Calculate the length of the average pufix for each prefix of s!

Input

The first line of input contains the number of test cases $z(1 \leq z \leq 10000)$. The descriptions of the test cases follow.

The first and only line of a test case contains the string s. The string consists of lowercase English letters and has length $l(1 \leq l \leq 1000000)$.

The total length of strings in all test cases does not exceed 5000000.

Output

For each test case print l integers in a single line. The i-th value should be equal to the length of the average pufix of the prefix of s of length i. If a given prefix has no pufixes, print -1 instead.

Example

standard input	standard output
2 aabaaab cdecdecde	-1 1 -1 1 1 1 3 -1 -1 -1 1 2 3 1 2 3

Note

In the first test case, the string `aabaaab` has only one pufix (`aab`), which has to also be its average pufix. Thus, the last value in the correct output (corresponding to the prefix of s comprising the entire string) is 3.

In the second test case, the string `cdecdecde` has two pufixes (`cde`, `cdecde`). Since the number of pufixes is even, the shorter one is chosen as the average pufix. Again, this corresponds to the last value in the correct output for this test case.

Problem H. Bookmarks

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

It's time for the adventure! Via Telegram, Innokenty just bought a north-oriented map of the sandbox in the next yard, where treasures can be found.

As presented on the map, the sandbox is a grid divided into $n \times m$ cells, aligned into n rows and m columns. From each cell, Innokenty can go to the neighboring edge cell. The sandbox can be entered in the top left corner (position $(1, 1)$ on the map) and exited in the bottom right corner (position (n, m) on the map). To avoid leaving many traces, **once Innokenty left a cell, he wouldn't return to it ever again**.

After studying the map for a little longer, Innokenty was able to distinguish three types of cells: treasuries — that is what Innokenty was looking for; dirt piles; and basic sand cells – nothing is there except sand. Innokenty quickly observed that treasuries and dirt piles are always surrounded by sand cells in exactly eight directions (horizontally, vertically, and diagonally). It also means that there are no treasuries and dirt piles at the boundary of the sandbox.

Innokenty wants to visit all the treasuries to collect all the "treasures" avoiding dirt piles.

Important business should not be left until the morning! Help Innokenty, Come up with a sequence of directions (`N`, `S`, `W`, `E`) you could take from the entry to the exit that visits all treasuries and steers clear of dirt. You may assume that a solution always exists.

Input

The first input line contains the number of test cases $z (1 \le z \le 2000)$. The descriptions of the test cases follow.

The first line of a test case contains two integers n and $m (3 \le n, m \le 1000)$ – the number of rows and columns of the map. Next n lines describe n rows of the map. Each line contains exactly m characters, each representing one room. Symbol ˙ (dot) indicates a hallway, `#` (hash) indicates a trap, and `$` (dollar) indicates a treasury.

The sum of the values of $n + m$ over all test cases does not exceed 20000.

Output

For each test case, print in a single line the directions Innokenty should follow.

Example

standard input	standard output
1 6 5 ..... .#.$. ..... .$... ...#.	SSEEENESSWWWWSSEEEE

Problem I. Summer and crossbows

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	256 megabytes

The dream of the most promising politician Zhenya finally came true — he took over Cowmos! The first thing he has to do, according to the justice march plan — is to restore justice among the deputies. Namely, he wants to increase their salaries so they will be equal.

Zhenya likes order and discipline, so he lined up all the n deputies in a line. Initially, the salary of the i-th deputy is a_i cigarettes. The only mathematical operation that Zhenya knows is multiplication. So that he would not be too mentally challenged, Zhenya decided to do only such operations: choose integer $k \geq 2$, choose k deputies standing consecutive, and multiply their salaries by k (Yeah, Zhenya doesn't shine with intelligence).

Zhenya wants to make all the deputies have equal salary, as long as it is at most 10^{18} cigarettes (because otherwise, there will be a lack of money and ammunition shortage of 70%). Given the initial salaries of deputies, determine if Zhenya's goal can be attained by applying some of the above-described operations. If so, determine the smallest number to which all deputies' salaries can be brought.

Input

The first input line contains the number of test cases $z(1 \leq z \leq 100\ 000)$. The descriptions of the test cases follow.

The first line of a test case contains an integer $n(2 \leq n \leq 1\ 000\ 000)$ – the number of deputies. The next line contains n integers $a_i(1 \leq a_i \leq 10^{18})$ — initial salaries of the deputies.

The total number of deputies in all test cases does not exceed 5 000 000.

Output

For each test case, if it's possible to use the growth spells to make all salaries equal to some number $m \leq 10^{18}$, then print the minimum such m, otherwise print -1, and Zhenya will turn back convoy to Belarus.

Example

standard input	standard output

3
4
22 22 11 33
4
20 40 40 20
7
120 20 5 5 20 5 5
	66
	-1
	120

Note

In the first test case, Zhenya can use operation with $k = 3$ on the first three deputies, followed by an operation with $k = 2$ on the last two, making all hedge heights equal to 66.

In the second test case, the first deputy's salary will remain shorter than the second after any number of operations, so the answer is -1. Note that Zhenya cannot decrease anyone's salary, otherwise, Zhenya will have his crossbow taken away from him.

In the final test case, Zhenya can make all salaries equal to 120 by using an operation with $k = 6$ once, followed by four operations with $k = 2$.

Problem J. Memory cut-out

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	256 megabytes

Oh no! Party went so well that you don't remember what happened last night! Now we have to remember all the adventures because the original plan was to have a lifetime of memories, but you forgot everything the next day!

Coming to your senses, you begin to realize that some memories are in the left hemisphere, and some are in the right hemisphere. To remember everything, you just need to combine these components into one whole.

Your left hemisphere is a $n_1 \times m_1$ grid of $n_1 \cdot m_1$ brain cells, from which some pieces have been cut out (consider them as memory gaps). By a piece, we mean a fragment of the grid, where every two cells are reachable from each other by moving through adjacent cells only. Two cells having only one common corner are not considered adjacent.

Some pieces of memory lie in your right hemisphere, that is, a $n_2 \times m_2$ grid. At first sight, they look like they are missing parts from your left hemisphere – but are they indeed? Decide whether the pieces match the cut-out (each memory piece from the right hemisphere must be matched with the corresponding memory gap in the left hemisphere).

Knowing yourself, you can assume that the following statements about your memory are true:

- Memory pieces do not have holes (fortunately for you);
- No two memory pieces are touching (even via corners) on both hemispheres;
- In the left hemisphere, the area formed by the cells not cut out is connected;
- In the right hemisphere, the area formed by the free cells is connected;
- There are at most 6 pieces cut out from the left hemisphere and at most 6 pieces laying in your right hemisphere;
- At least one memory piece has been cut out from the left, and there is at least one memory piece in the right hemisphere;
- Each memory piece fits in a 10×10 bounding box;
- Memory pieces cannot be rotated or flipped.

Input

The first line of input contains the number of test cases $z (1 \le z \le 2000)$. The descriptions of the test cases follow.

The first line of each test case contains two integers $n_1, m_1 (1 \le n_1, m_1 \le 64)$ — the number of rows and columns of your left hemisphere. Next, n_1 lines describe the grid — each line contains m_1 characters. The character `x` means the cell has not been cut out. The character `.` denotes a hole in your memory.

The following line contains two integers $n_2, m_2 (1 \le n_2, m_2 \le 64)$ — the number of rows and columns of your right hemisphere. Next n_2 lines describe the grid in it — each line contains m_2 characters. The character `x` means this cell belongs to a piece. The character `.` denotes a free cell.

Output

For each test case, print in a single line `YES` if the pieces match, and you will be able to recover your memories. Otherwise, output `NO` and call your friends; maybe they'll remember something...

Example

standard input	standard output
2 5 13 xxx..xxxxxxxx x....xxxx.xxx x..xxxxx...xx xxxxxxxxx.xxx xxxxxxxxxxxxx 7 9 ....x.... ...xxx... ....x.... ....xx... ..xxxx... ..xx..... 5 13 xxx..xxxxxxxx x....xxxx.xxx x..xxxxx...xx xxxxxxxxx.xxx xxxxxxxxxxxxx 7 9 ....x.... ...xxx... ....x.... ....xx..... ..xxxx... ....xx...	YES NO