Problem A. Kingdoms

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

Several kingdoms got into serious financial troubles. For many years, they have been secretly borrowing more and more money from each other. Now, with their liabilities exposed, the crash is inevitable...

There are n kingdoms. For each pair (A, B) of kingdoms, the amount of gold that kingdom A owes to kingdom B is expressed by an integer number d_{AB} (we assume that $d_{BA} = -d_{AB}$). If a kingdom has negative balance (has to pay more than it can receive), it may bankrupt. Bankruptcy removes all liabilities, both positive and negative, as if the kingdom ceased to exist. The next kingdom may then bankrupt, and so on, until all remaining kingdoms are financially stable.

Depending on who falls first, different scenarios may occur — in particular, sometimes only one kingdom might remain. Determine, for every kingdom, whether it can become the only survivor.

Input

The first line of the input contains the number of test cases $T \leq 10^4$. The descriptions of the test cases follow:

The description of each test case starts with a line containing the number of the kingdoms n, $1 \leq n \leq 20$. Then n lines follow, each containing n space-separated numbers. The j-th number in the i-th line is the number d_{ij} of gold coins that the i-th kingdom owes to the j-th one. You may assume that $d_{ii} = 0$ and $d_{ij} = -d_{ji}$ for every $1 \leq i, j \leq n$. Also, $|d_{ij}| \leq 10^6$ for all possible i, j.

It is guaranteed that total sum of all n in one test case does not exceed $5 \cdot 10^4$.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case, print a single line containing the indices of the kingdoms that can become the sole survivors, in increasing order. If there are no such kingdoms, print a single number 0.

Example

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

Problem B. Who wants to live forever? (Division 1 Only!)

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

Digital physics is a set of ideas and hypotheses that revolve around the concept of computable universe. Maybe our universe is just a big program running on a Turing machine? Is the state of the universe finite? Will the life of the universe end? We can only theorize.

In order to help advance the current state of knowledge on digital physics, we ask you to consider a particular model of the universe (which we shall call Bitverse) and determine whether its life comes to a conclusion or continues evolving forever.

Bitverse consists of a single sequence of n bits (zeros or ones). The universe emerges as a particular sequence, in an event called the "Bit Bang", and since then evolves in discrete steps. The rule is simple — to determine the next value of the i-th bit, look at the current value of the bits at positions $i-1$ and $i+1$ (if they exist; otherwise assume them to be 0). If you see exactly one 1, then the next value of the i-th bit is 1, otherwise it is 0. All the bits change at once, so the new values in the next state depend only on the values in the previous state. We consider the universe dead if it contains only zeros.

Given the state of the universe at the Bit Bang, answer the following fundamental question: will Bitverse live forever, or will it eventually die?

Input

The first line of the input contains the number of test cases $T \leq 65535$. The descriptions of the test cases follow:

Each test case is a string of at least 1 and at most $2 \cdot 10^5$ characters '`0`' or '`1`'.

Total size of input file does not exceed 10 mebibytes.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case, print "`LIVES`" if the universe lives forever, and "`DIES`" otherwise.

Example

standard input	standard output
3 01 0010100 11011	LIVES DIES LIVES

Note

The first example universe will never become a sequence of zeros (it will continue flipping: 01 10 01 ...). The second one will die in a few steps (0010100 0100010 1010101 0000000). The third one does not change.

Problem C. Chemist's vows

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

Chemist Clara swore a solemn vow — from now on, she can only speak atomic element symbols. Of course, this limits her ability to talk. She can say, for example, "`I Am CLaRa`" (as I is the symbol of iodine, Am is americium, C is carbon and so on). She can also say "`InTeRnAtIONAl`", but she has a lot of trouble with "`collegiate`", "`programming`" and "`contest`".

Given a word, determine whether Clara can speak it (i.e. if it is a concatenation of atomic symbols). Without your help, she might as well have taken silence vows!

You may identify upper- and lowercase letters, as Clara cannot speak uppercase anyway. In case you forgot the elements' symbols, here is the complete periodic table:

H																	He
Li	Be											B	C	N	O	F	Ne
Na	Mg											Al	Si	P	S	Cl	Ar
K	Ca	Sc	Ti	V	Cr	Mn	Fe	Co	Ni	Cu	Zn	Ga	Ge	As	Se	Br	Kr
Rb	Sr	Y	Zr	Nb	Mo	Tc	Ru	Rh	Pd	Ag	Cd	In	Sn	Sb	Te	I	Xe
Cs	Ba	*	Hf	Ta	W	Re	Os	Ir	Pt	Au	Hg	Tl	Pb	Bi	Po	At	Rn
Fr	Ra	**	Rf	Db	Sg	Bh	Hs	Mt	Ds	Rg	Cn		Fl		Lv		

*	La	Ce	Pr	Nd	Pm	Sm	Eu	Gd	Tb	Dy	Ho	Er	Tm	Yb	Lu
**	Ac	Th	Pa	U	Np	Pu	Am	Cm	Bk	Cf	Es	Fm	Md	No	Lr

Input

The first line of the input contains the number of test cases $T \le 10^4$. The descriptions of the test cases follow:

Each test case is a single lowercase word over the English alphabet. The length of the word is positive and does not exceed $5 \cdot 10^4$.

Total size of input file does not exceed $6 \cdot 10^5$ bytes.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case print a single line containing the word "`YES`" if Clara can say the given word, and "`NO`" otherwise.

Example

standard input	standard output
4 international collegiate programming contest	YES NO NO NO

Note

On `http://acm.math.spbu.ru:17249/files/opencup/oc12/gp6/c.tex` there is a TeX version of this problem statement.

Problem D. Non-boring sequences

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

We were afraid of making this problem statement too boring, so we decided to keep it short.

A sequence is called *non-boring* if its every connected subsequence contains a unique element, i.e. an element such that no other element of that subsequence has the same value (for a string $(a_1, a_2, ..., a_n)$ a connected subsequence is a string $(a_i, a_i + 1, ..., a_j)$ for some $1 \le i \le j \le n$).

Given a sequence of integers, decide whether it is *non-boring*.

Input

The first line of the input contains the number of test cases T. The descriptions of the test cases follow:

Each test case starts with an integer n ($1 \le n \le 2 \cdot 10^5$) denoting the length of the sequence. In the next line the n elements of the sequence follow, separated with single spaces. The elements are non-negative integers less than 10^9.

Sum of all n in input file does not exceed $1.2 \cdot 10^6$.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case print a single line containing the word "`non-boring`" or "`boring`".

Example

standard input	standard output
4 5 1 2 3 4 5 5 1 1 1 1 1 5 1 2 3 2 1 5 1 1 2 1 1	non-boring boring non-boring boring

Problem E. Word equations

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

You are given a text T and a pattern P. You want to check if you can erase some of the letters of T so that the remaining symbols produce exactly P. For example, the word "programming" can be partially erased to obtain "pong" or "program" or "roaming" (but not "map", as the letters must remain in the same order). Both words consist of small letters of the English alphabet only.

There is just one catch: the text T is encoded by a system of equations. The equations use special symbols (every symbol is denoted by a word in capital letters), each of them encoding some word over the alphabet $\{a, \ldots, z\}$. Each equation is of one of the following forms:

$$A = a\ word\ over * \{a, \ldots, z\}$$

or

$$A = B + C$$

where A, B, C can be any special symbols (note that B and C can be the same), and the `+` sign denotes the concatenation of words (both `+` and `=` are surrounded by spaces. The system is:

- unambiguous — for a fixed symbol A, there is exactly one equation with A on the left-hand side, and
- acyclic — if you start from any symbol A and make substitutions according to the equations (right-hand side for left-hand side), you can never obtain an expression containing A again.

Such a system always has a unique solution. For example, in the system:

```
START = FIRST + SECND

FIRST = D + E

SECND = F + E

D = good

E = times

F = bad
```

the symbol START encodes the word "`goodtimesbadtimes`".

Given a single word P as the pattern, a system of equations, and one particular starting symbol S of this system, determine whether the pattern P is present in the word encoded by S.

Input

The first line of the input contains the number of test cases $T \leq 500$. The descriptions of the test cases follow:

Each test case starts with a line containing a single integer k ($1 \leq k \leq 500$) — the number of equations. The next k lines contain equations. Each of them has one of the two forms given in the problem statement, with spaces separating words, plus signs, and equation signs. Each word (including symbol names) is at least one and at most five characters long.

The next line contains a single special symbol (a word in capital letters), while the final line contains a non-empty word of at most 2 000 lowercase letters. These are the starting symbol and the pattern to find, respectively.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case print the answer in a separate line: "YES" if the pattern appears in the given encoded word, and "NO" otherwise.

Example

standard input	standard output
1 6 START = FIRST + SECND FIRST = D + E SECND = F + E D = good E = times F = bad START debate	YES

Problem F. Farm and factory (Division 1 Only!)

Input file:	`standard input`
Output file:	`standard output`
Time limit:	15 seconds
Memory limit:	256 mebibytes

King Bitolomew thinks that Byteland is a rather unique country. It is quite small, and all of its citizens (excluding the king) work either at the farm or at the factory, which are located in two distinct cities. So, every morning, the inhabitants of each city commute to these two cities in great traffic jams.

The road network of Byteland consists of undirected roads joining pairs of distinct cities. The roads do not cross outside cities (but tunnels and bridges may occur). There may exist multiple direct roads between two cities. The farm and the factory are both reachable from all cities.

A few months ago, in an attempt to improve the traffic situation, king Bitolomew has introduced tolls on every road, requiring citizens to pay a fixed (per road) amount every time they want to use a road. Bitolomew hoped that the prospect of paying money would force some citizens to reconsider their routes, and thus distribute the traffic more evenly.

The king's idea turned out to be, as his advisors say, less than perfect. Every citizen of Byteland now uses the cheapest route to commute to work! King Bitolomew is not overly concerned about this, as the income from tolls has really improved the kingdom's budget. In fact, the king's finances are now so good that he plans to build himself a new capital with a new castle in it. This new capital should be connected with some other cities by direct roads, so that every city is reachable from it. The newly created roads can have any non-negative tolls assigned (in particular, the tolls do not have to be integer).

King Bitolomew really dislikes the noise generated by cars passing by his castle. He would like to set the tolls for the new roads going out from his new capital so that for any city v other than the capital there exist cheapest paths from v to both the farm and the factory that do not pass through the capital (note that v here can also be the city with the farm or the factory). On the other hand, since the king is not exempt from the tolls, he would like to minimize the average cost of cheapest paths from the new capital to every other city.

Help the king determine that minimum possible cost.

Input

The first line of the input contains the number of test cases $T \leq 1.1 \cdot 10^4$. The descriptions of the test cases follow:

Each test case starts with two integers n, m ($2 \leq n \leq 10^5,\ 1 \leq m \leq 3 \cdot 10^5$) denoting the number of cities and the number of roads in Byteland, respectively. The next m lines describe the roads. Each road is described by three integers u, v, c ($1 \leq u, v \leq n, u \neq v,\ 0 \leq c \leq 10^6$), denoting the indices of the two cities joined by the road and the toll the king has set for that road. There may be multiple roads between any given pair of cities.

The indices of the cities with the farm and the factory are 1 and 2, respectively.

Total sum of n in one test file does not exceed $6 \cdot 10^5$.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case print a single line containing the minimum possible average cost of reaching other cities from the newly created capital. Your answer will be accepted if its absolute or relative error is below 10^{-8}.

Example

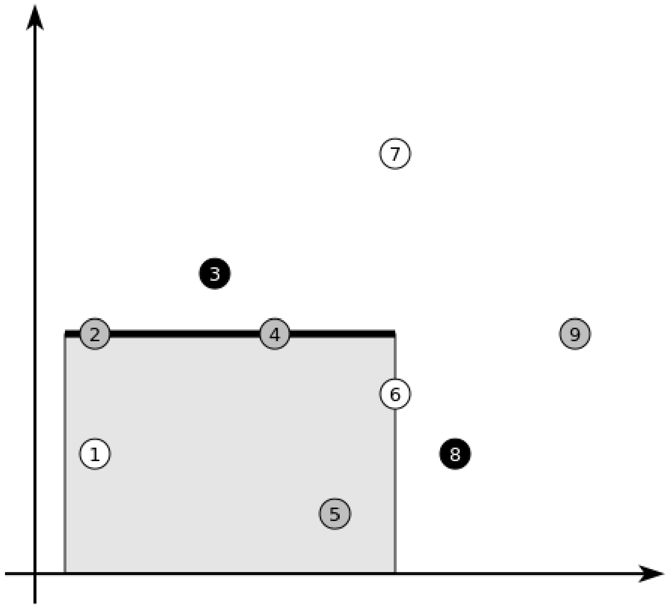
standard input	standard output
1 3 3 1 2 5 2 3 5 3 1 1	1.833333333333

Problem G. Jewel Heist (Division 1 Only!)

Input file:	`standard input`
Output file:	`standard output`
Time limit:	15 seconds
Memory limit:	256 mebibytes

Arsen Lupin, the master thief, aims to steal Evil Erwin's jewels. Erwin has n jewels on display in his store. Every precious stone is of one of k distinct colors. The exposition is so large that we can treat it as the Euclidean plane with the jewels being distinct points. The display is secured by some quite expensive alarms.

Lupin has invented a device: a big, robotic hand that can grab some of the Erwin's jewels without triggering any of the alarms. The hand can make one (and only one) grab, taking all the jewels lying on some horizontal segment or below it (see the picture). Lupin could easily take all the jewels this way, but he knows that the more he takes, the harder it will be to get rid of them. He decided that the safest way is to take a set of jewels that does not contain all the k colors.



The robotic hand grabs jewels 1,2,4,5 and 6, carefully omitting the black ones

Compute how many jewels Lupin can steal with one grab of his device, without taking jewels in every color.

Input

The first line of the input contains the number of test cases $T \leq 200$. The descriptions of the test cases follow:

Each test case starts with two integers n ($2 \leq n \leq 2 \cdot 10^5$) and k ($2 \leq k \leq n$) denoting the number of jewels and the number of distinct colors. The next n lines denote the jewels' positions and colors. The j-th line contains three space-separated integers x_j, y_j, c_j ($1 \leq x_j, y_j \leq 10^9$, $1 \leq c_j \leq k$) meaning that the j-th jewel lies at coordinates (x_j, y_j) and has color c_j.

You may assume that there is at least one stone of every color at the exposition.

Sum of all n in input file does not exceed 10^6.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case print a single line containing the maximum possible number of stolen jewels.

Example

<table>
<tr><th>standard input</th><th>standard output</th></tr>
<tr><td><pre>1
10 3
1 2 3
2 1 1
2 4 2
3 5 3
4 4 2
5 1 2
6 3 1
6 7 1
7 2 3
9 4 2</pre></td><td><pre>5</pre></td></tr>
</table>

Problem H. Darts

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

Consider a game in which darts are thrown at a board. The board is formed by 10 circles with radii 20, 40, 60, 80, 100, 120, 140, 160, 180, and 200 (measured in millimeters), centered at the origin. Each throw is evaluated depending on where the dart hits the board. p points ($p \in \{1, 2, \ldots, 10\}$) if the smallest circle enclosing or passing through the hit point is the one with radius $20 \cdot (11 - p)$. No points are awarded for a throw that misses the largest circle. Your task is to compute the total score of a series of n throws.

Input

The first line of the input contains the number of test cases $T \leq 10^4$. The descriptions of the test cases follow:

Each test case starts with a line containing the number of throws n ($1 \leq n \leq 10^6$). Each of the next n lines contains two integers x and y ($-200 \leq x, y \leq 200$) separated by a space — the coordinates of the point hit by a throw.

Sum of all n in one input file does not exceed $1.1 \cdot 10^6$.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case print a single line containing one integer — the sum of the scores of all n throws.

Example

standard input	standard output
1 5 32 -39 71 89 -60 80 0 0 196 89	29

Problem I. The Dragon and the knights

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

The Dragon of the Wawel Castle, following the conflict with the local Shoemakers' Guild, decided to move its hunting grounds out of Krakow, to a less hostile neighborhood. Now it is bringing havoc and terror to the peaceful and serene Kingdom of Bytes.

In the Kingdom of Bytes there are n rivers and each of them flows along a straight line (that is, you may think of the Kingdom as the Euclidean plane divided by infinite lines). Rivers may be parallel. No three rivers have a common point. The rivers divide the Kingdom into some *districts*.

Fortunately, there are m valiant knights in the Kingdom. Each of them has taken his post and swore an oath to protect his district. The Kingdom is thus protected for evermore... or is it?

It is known that Dragon will not attack a district which has at least one knight inside. The knights, however, are famous for their courage in battle, not for their intelligence. They may have forgotten to protect some of the districts.

Given a map of the Kingdom and the knights' positions, determine whether all districts are protected.

Input

The first line of the input contains the number of test cases $T \leq 75$. The descriptions of the test cases follow:

Each test case starts with a line containing the number of rivers n ($1 \leq n \leq 100$) and the number of knights m ($1 \leq m \leq 5 \cdot 10^4$). Then follow n lines describing rivers. The j-th of them contains three space-separated integers A_j, B_j, C_j of absolute values not exceeding 10^4. These integers are the coefficients of the equation $A_j \cdot x + B_j \cdot y + C_j = 0$ of the line along which the j-th river flows. After that, there are m lines describing the positions of the knights: the i-th of these lines contains two integers X_i, Y_i ($-10^9 \leq X_i, Y_i \leq 10^9$) — the coordinates of the i-th knight. You may assume that no knight is standing in a river (his shining armour would quickly rust if he did). Two knights may occupy the same post (their coordinates can be equal). No two rivers flow along the same line and no three rivers have a common point.

Sum of all m and n in one input file does not exceed $6 \cdot 10^5$.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case, output a single line containing a single word "`PROTECTED`" if all districts are safe from the Dragon, and "`VULNERABLE`" otherwise.

Example

standard input	standard output
2 3 7 0 1 0 1 0 0 1 1 -3 1 1 5 -1 3 2 2 -2 -2 6 -1 -2 -8 4 1 1 0 1 0 0 1	PROTECTED VULNERABLE

Problem J. Conservation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	256 mebibytes

The most famous painting in Byteland — a portrait of a lady with a computer mouse by Leonardo da Bitci — needs to be conserved. The work will be conducted in two narrowly specialized laboratories. The conservation process has been divided into several stages. For each of them, we know the laboratory in which it will take place.

Transporting the very precious and fragile painting introduces additional risk; therefore, it should be avoided whenever possible. Ideally, all the work in the first laboratory would be done, and then the painting would be moved to the second one. Unfortunately, there are several dependencies between the conservation stages — some of them need to be completed before others may begin.

Your task is to find an ordering of conservation stages that minimizes the number of times the painting needs to be moved from one laboratory to the other. The conservation can begin or end in any of the two laboratories.

Input

The first line of the input contains the number of test cases T. The descriptions of the test cases follow:

The first line of each test case contains two space-separated integers n and m ($1 \leq n \leq 10^5$, $0 \leq m \leq 10^6$) — the number of conservation stages and the number of dependencies between them.

In the next line there are n space-separated integers — the i-th of them is 1 if the i-th conservation stage will take place in the first laboratory, and 2 otherwise. The following m lines contain pairs of integers i, j ($1 \leq i, j \leq n$), denoting that the i-th stage has to be completed before the j-th.

You may assume that it is always possible to order the conservation stages so that all the dependencies are satisfied.

Sum of all n in input file does not exceed $5 \cdot 10^5$. Sum of all m in input file does not exceed $3 \cdot 10^6$.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case, output a single line containing the minimal number of times the painting needs to be transported between the laboratories. If all work can be done by one laboratory, print 0.

Example

standard input	standard output
1 5 6 1 2 1 2 1 1 2 1 3 2 4 3 4 2 5 3 5	2

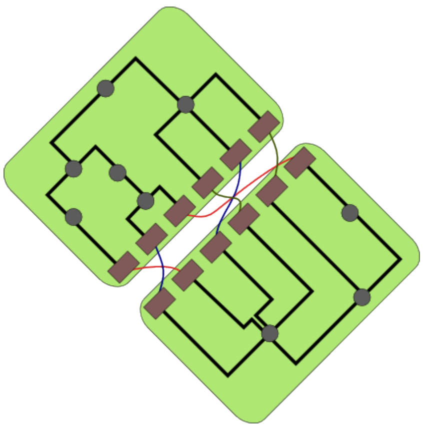
Problem K. Graphic Madness

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

In Byteland, there are two leading video card manufacturers: Bitotronics and 3D-Bytes. Their top-of-the-line cards are quite similar. Each of them consists of many *nodes*, connected with wires transferring the signal that is being processed. The products contain two kinds of nodes: sockets and processors. The wire network fulfills the following conditions:

- Each socket is connected to exactly one processor and no other sockets.
- Each processor is connected to at least two other nodes.
- For any two nodes in the network, there is exactly one path of wires connecting them. In other words, the graph of connections between nodes is a tree.

Bitthew loves to tinker with computer hardware. He has bought two video cards, one from each manufacturer. Since accidentally the cards have the same number of sockets, he has decided to connect each socket of the Bitotronics card to a distinct socket of the 3D-Bytes card with cables. The device he obtained looks like this:



Bitthew would like to squeeze out maximum processing power from the device. In order to do that, he wants to find a path through wires and cables that the processed signal can take. The path should visit each node of both cards exactly once, and it should start and end at the same node on the same card. Help Bitthew find out whether this can be done.

Input

The first line of the input contains the number of test cases $T \leq 35$. The descriptions of the test cases follow:

Each test case starts with three integers k, n, m ($2 \leq k \leq 1000$, $1 \leq n, m \leq 1000$) denoting respectively the number of sockets on each card, the number of processors on the Bitotronics card, and the number of processors on the 3D-Bytes card. The nodes on the cards are named as follows:

- the sockets on the Bitotronics card: AS1, AS2, $\ldots$, ASk

- the processors on the Bitotronics card: AP1, AP2, ..., APn
- the sockets on the 3D-Bytes card: BS1, BS2, ..., BSk
- the processors on the 3D-Bytes card: BP1, BP2, ..., BPm

The next $n+k-1$ lines contain the description of the wire network on the Bitotronics card. Each of these lines contains the names of two different nodes on that card that are connected directly by a wire. The description is followed by a blank line. The next $m+k-1$ lines contain the description of the wire network on the 3D-Bytes card in the same format. The description is followed by another blank line. The last k lines of the test case describe the cables added by Bitthew. Each of these lines contains the names of two sockets on different cards that are directly connected by a cable. Every socket will be present on exactly one of these k lines. There is a blank line after each test case except the last one.

Output

Print the answers to the test cases in the order in which they appear in the input. For each test case print a single line containing the answer. If there exists no path with the desired properties, output "NO". Otherwise, output "YES" followed by a description of the path: $n + m + 2k$ distinct nodes in the order in which the signal will pass through them. Every two consecutive nodes should be connected by a wire or a cable. Additionally, the first and the last node must be connected.

Example

standard input

```
1
2 1 11
AS1 AP1
AS2 AP1

BS1 BP1
BS2 BP11
BP1 BP2
BP2 BP3
BP3 BP4
BP4 BP5
BP5 BP6
BP6 BP7
BP7 BP8
BP8 BP9
BP9 BP10
BP10 BP11

AS1 BS2
BS1 AS2
```

standard output

```
YES BP11 BP10 BP9 BP8 BP7 BP6 BP5 BP4 BP3 BP2 BP1 BS1 AS2 AP1 AS1 BS2
```

Problem L. User name (Division 2 Only!)

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

An operating system PiOS assigns user names according to the following set of rules:

1. The maximum length of a username is MAXLEN characters. (The value of MAXLEN will be specified in the input for each problem instance.)
2. The first character of the user name is the first letter of the person's first name, converted to lower case. Ignore apostrophes and hyphens here and in Step 3.
3. Append as many letters of the person's last name as possible (converted to lower case, if necessary), without exceeding a total of MAXLEN characters. Starting with the first letter of the last name, append these letters in the order in whch they appear in the last name.
4. If a user name assigned on basis of Rules $1 - 3$ already exists in the database, break the tie as follows: append serial numbers $1 - 9$, in that order, to the username from step 3, if that can be done without exceeding the limit of MAXLEN characters in the username. Otherwise, drop the last letter before appending the serial number.
5. If a user name assigned on basis of Rules $1 - 4$ already exists in the database, break the tie as follows: append serial numbers $10 - 99$, in that order, to the username from step 3, if that can be done without exceeding the limit of MAXLEN characters in the username. Otherwise, drop the last letter or the last two letters (whichever is necessary) before appending the serial number.
6. It is assumed that the above rules will avoid ties.

Input

The input will contain data for a number of test cases. The first line of each test case will contain two positive integers: the number of names and the value of MAXLEN ($5 \leq MAXLEN \leq 80$). This will be followed by the list of names. Each name will consist of at most 80 characters and will begin with the first name, followed by middle names, if any, and will conclude with the last name. A single blank space will separate first, middle, and last names. Any name can contain upper and lower case letters, hyphens, and apostrophes. A last name will contain at least two letters, other names will contain at least one letter (they could be just initials). There will be no more than 200 names in each case. The last test case will be followed by a line containing two zeros for the number of names and MAXLEN.

Output

For each case, the output will begin with a line containing the case number. This will be followed by the list of user names, one per line, in the same order as the corresponding names in the input.

Example

standard input	standard output
2 6	Case 1
John Ax	jax
Christos H Papadopulos	cpapad
11 8	Case 2
Jean-Louis d'Arnoux	jdarnoux
Jean-Louis A d'Arnoux	jdarnou1
Jean-Louis B d'Arnoux	jdarnou2
Jean-Louis C d'Arnoux	jdarnou3
Jean-Louis D d'Arnoux	jdarnou4
Jean-Louis D d'Arnoux	jdarnou5
Jean-Louis F d'Arnoux	jdarnou6
Jean-Louis G d'Arnoux	jdarnou7
Jean-Louis H d'Arnoux	jdarnou8
Jean-Louis I d'Arnoux	jdarnou9
Jean-Louis J d'Arnoux	jdarno10
11 9	Case 3
Jean-Louis d'Arnoux	jdarnoux
Jean-Louis A d'Arnoux	jdarnoux1
Jean-Louis B d'Arnoux	jdarnoux2
Jean-Louis C d'Arnoux	jdarnoux3
Jean-Louis D d'Arnoux	jdarnoux4
Jean-Louis D d'Arnoux	jdarnoux5
Jean-Louis F d'Arnoux	jdarnoux6
Jean-Louis G d'Arnoux	jdarnoux7
Jean-Louis H d'Arnoux	jdarnoux8
Jean-Louis I d'Arnoux	jdarnoux9
Jean-Louis J d'Arnoux	jdarnou10
0 0	

Problem M. King (Division 2 Only!)

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

Your task is to help the King to find the shortest way to his capital.

For simplicity, we will assume that kingdom is a rectangular grid consisting of unit squares. For now, he stays at some square and capital is at some other square.

As King, he prefers to move from one square to the other by next way: each move takes him to the adjacent square in any of 8 possible directions.

Input

The input consists of several (less than $15 \cdot 10^3$) test cases, each of them specified by six integers on one line: X, Y, K_X , K_Y , C_X , and C_Y . X and Y specify the size of the grid in unit squares, $1 \le R, C \le 100$.

King cannot leave his kingdom, because it would be too dangerous. The values of K_X , K_Y are the coordinates of the square that King is standing on, and C_X, C_Y are the coordinates of the square with castle ($1 \le K_X, C_X \le X$; $1 \le K_Y, C_Y \le Y$).

Output

For each test case, print one integer — the minimal number of moves that King needs to reach his castle.

Example

standard input	standard output

`10 10 10 10 1 1`
`2 2 1 1 1 2`
`5 5 3 3 5 5`
`6 6 1 1 1 1`
	`9`
	`1`
	`2`
	`0`

Problem N. Nonogram (Division 2 Only!)

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

A Nonogram is a pencil puzzle played on a grid. The grid is initially blank. There are numbers on the side and top of the grid, which indicate how the grid squares should be filled in. The numbers measure how many unbroken lines of filled-in squares there are in any given row or column. For example, a clue of "4 8 3" would mean there are sets of four, eight, and three filled squares, in that order, with at least one blank square between successive groups.

You are going to work backwards. Given a Nonogram solution, produce the numbers which should be at the side and top of the grid.

Input

There will be several test cases in the input file. Each test case will begin with an integer n ($2 \le n \le 100$) indicating the size of the grid. Each of the next n lines will have exactly n characters, consisting of either '`.`' for a blank square, or '`X`' for a square which has been filled in. The input will end with a line with a single 0.

It is guaranteed that input file contains no more than 1500 lines.

Output

For each test case, print $2n$ lines of output. The first n lines represent the numbers for the rows, from top to bottom. The next n lines represent the numbers across the top, from left to right. If any row or column has no squares filled in, output a 0.

Example

standard input	standard output
3	3
XXX	2
.XX	1
.X.	1
3	3
X.X	2
..X	1 1
X..	1
5	1
..X..	1 1
.XXX.	0
X.X.X	2
..X..	1
..X..	3
0	1 1 1
	1
	1
	1
	1
	5
	1
	1