Problem A. Area and Perimeter

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	128 megabytes

You are given a positive irreducible fraction p/q. Your task is to either find a connected figure from cells whose area to perimeter ratio is exactly p/q, or report that such a figure does not exist.

Input

The first line contains a single positive integer T — the number of testcases ($1 \le T \le 10$).

Each of the following T lines contains a positive irreducible fraction p/q ($1 \le p, q \le 20$).

Output

For each testcase print:

- "NO" on a separate line if there is no solution for the given fraction p/q.
- "YES" on a separate line if there is an solution. On the next line print space-separated integers n and m ($1 \le n, m \le 100$), and on the next n lines print m characters each — a description of an example of the figure. Use "#" to indicate the cell that belongs to the figure, for other cells use ".". The cells of the figure should form a connected shape. If there are several solutions — print any. See examples for more precise understanding.

It can be proved that if for some fraction from the input the desired figure exists, then there also exists one that fits into the region of size 100 by 100.

Example

standard input	standard output
5 1/4 3/8 2/7 1/2 3/2	YES 1 1 # YES 3 4 .... .##. .#.. NO YES 3 3 ### #.# ### YES 8 8 ...####. .######. #######. #######. ######## .####### ...##### ....####

Problem B. Boarding

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	128 megabytes

The airline company "Berland Airlines" has recently started offering its customers a new service. For a small additional fee, a passenger can check in electronically for a flight instead of queuing at the check-in desk at the airport. In this case, the passenger can choose a seat in the aircraft cabin in advance. At the check-in counter, the seats are distributed randomly from those that were not occupied by passengers who passed electronic registration.

As you know, many people travel in groups. The airline company knows each group because, for example, when buying tickets via the Internet, these groups of people make a purchase with one order for several people. The company's management wants to gently motivate customers to use the new service. To this end, it was decided to distribute seats in the cabin for those customers who refused electronic registration in such a way that no two people from any group would sit side by side.

Seats in the cabin are arranged in n rows, each of which has 6 seats labeled A, B, C, D, E and F from left to right. Between places C and D there is a passage that crosses all rows of the seats. Thus, a pair of places that are located in the same row and are labeled with one of the pairs (A, B), (B, C), (D, E), or (E, F) are considered neighbour.

You know about each place whether it is occupied by a client who has passed electronic registration or not. You also know the number of client groups k that refused electronic registration, for each group you know the number of people in the group a_i ($1 \le i \le k$). Your task is to distribute these passengers in such a way that any two people from the same group sit in seats that are not neighbour. Or determine that such a seating arrangement is impossible.

Input

The first line contains an integer n ($1 \le n \le 50$).

Then there are n lines describing the seats in the aircraft cabin. Each of these lines is of the form "*ABC DEF*", where the letters from A to F are from the set {"`X`", "`.`"}, where "`X`" means that the corresponding seat is occupied, "`.`" — it is free.

The next line contains an integer k ($1 \le k \le 26$).

The last line contains k integers a_i ($1 \le a_i \le 6n$, $1 \le i \le k$).

It is guaranteed that $\sum a_i$ does not exceed the number of unoccupied seats.

Output

In the first line print "`YES`" or "`NO`" (without quotes) depending on whether the desired seating arrangement exists or not.

If the seating arrangement exists — additionally output n lines: the desired seating arrangement. These n lines should repeat the description of the aircraft cabin seats from the input, where some characters "`.`" should be replaced with small latin letters, which determine the group of the person occupied this seat. For the first group, use "`a`", for the second one — "`b`", and so on. The total number of "`a`" letters used should be a_1, the total number of "`b`" letters — a_2, and so on. No two identical letters should be adjacent. See examples for more precise understanding.

If there are several solutions — print any of them.

Examples

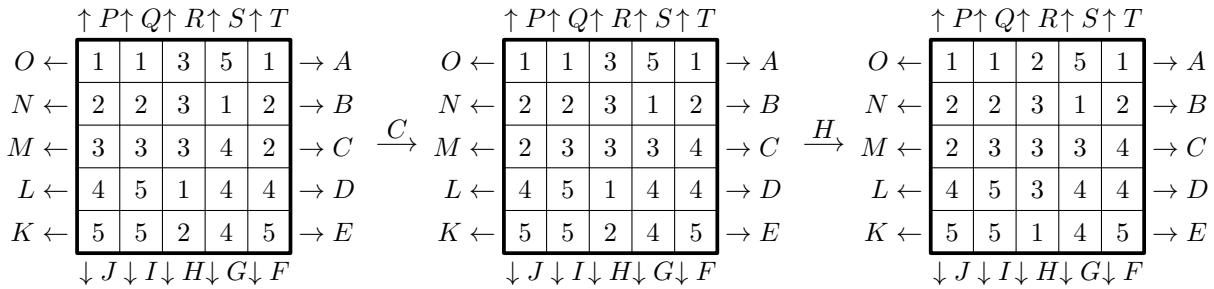
standard input	standard output
3 ... ... X.. .X. 4 6 5 2 1	YES aba aba acb acb Xdb .X.
3 ... ... .XX ... X.. ..X 2 9 2	YES aba aba aXX a.a X.a a.X
3 ... ... .XX ... X.. ..X 2 2 10	NO

Problem C. Cross Puzzle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	128 megabytes

You are given a table of numbers of size 5×5, each cell of the table contains an integer from 1 to 5, and each number from 1 to 5 occurs in the table exactly 5 times.

In one move, it is allowed to cyclically shift all the numbers in any row or in any column. Each possible move is represented by a symbol from "`A`" to "`T`", as shown below. The goal of the game is to collect a cross of identical numbers in the center of the table in the least number of moves.



Write a program that finds the shortest sequence of moves that solves the puzzle. If there are several such sequences — find the lexicographically smallest one.

Input

The first line contains an integer T — the number of tests ($1 \le T \le 10$). Then there are $5T$ lines with 5 numbers each. The first 5 of them describe the first test, the next 5 describe the second one, and so on. It is guaranteed that every test uses each of the numbers from 1 to 5 exactly 5 times.

Output

Print T lines — one line per test. For each test, print the shortest sequence of moves as a string consisting of letters from "`A`" to "`T`". If there are several the shortest sequences — print the lexicographically smallest one. If the puzzle is already complete, instead of an empty string, print "`Already done`" without quotes. If the puzzle cannot be solved — print "`Impossible`" without quotes.

Example

standard input	standard output
3 1 1 3 5 1 2 2 3 1 2 3 3 3 4 2 4 5 1 4 4 5 5 2 4 5 1 1 2 5 1 2 2 3 1 2 2 3 3 3 4 4 5 3 4 4 5 5 1 4 5 2 3 4 5 2 5 1 1 3 4 1 4 2 4 5 1 3 1 3 4 2 5 3 5 2	CH Already done ICPC

Problem D. Deleting Numbers

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	128 megabytes

You are given an array of positive integers of length n. All its elements are numbered by consecutive numbers from 1 to n from left to right. All numbers in the array are pairwise distinct.

You have to remove the largest amount of numbers from the array. You can delete a number only if its parity matches the parity of the position in the array where this number is located. After deletion, all numbers to the right of the deleted one are shifted to the left by 1, and the length of the array is reduced by one. Note that after deleting a number, the ability to delete other numbers in the array may appear or disappear.

For example, let the initial array is $[2, 8, 7, 9, 5]$. The numbers 8, 7, and 5 can be removed. Let's remove the number 8. Then the array will take the form $[2, 7, 9, 5]$. Now only the number 9 can be removed. Let's delete it, the array will become $[2, 7, 5]$. Now only 5 can be deleted, after deleting which we get the array $[2, 7]$. Further numbers cannot be deleted. Note that for the array $[2, 8, 7, 9, 5]$ it is possible to carry out such a sequence of deletions that only one number remains in the end.

Determine in what order the numbers should be removed in order to remove the maximum number of them.

Input

The first line contains an integer n ($1 \le n \le 10^5$).

The second line contains n positive integers separated by a space — the initial array.

The numbers in the array are pairwise distinct and do not exceed 10^9.

Output

In the first line print an integer k — the maximum amount of numbers that can be removed ($0 \le k \le n$). In the second line print k numbers from the input array separated by a space in the order in which they need to be removed. If there are several optimal deletion sequences — print any of them.

Example

standard input	standard output
5 2 8 7 9 5	4 7 9 5 8

Problem E. Easy Moving

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	128 megabytes

One day, Alice sent Bob the string s, consisting of the letters "E", "N", "S", "W" and the following message:

"If you imagine a Cartesian coordinate system, where the x axis points to the east, and y to the north, then stand at the point $(0, 0)$ and sequentially execute all commands from the string s from left to right (where "E" — move 1 step to the east, "N" — move 1 step to the north, "S" — 1 step to the south, "W" — 1 step to the west), then you come to the point where the treasure is buried".

Without thinking twice, Bob executed all the commands from the string s and got to the point (x_0, y_0). He did not find the treasure there, which he informed Alice about. To which Alice replied:

"Oh, I made a mistake. In the string s, replace the i-th letter with c. And then the string s will definitely lead to the treasure".

Bob replaced the i-th letter in s with c, repeated all the commands again, got to the point (x_1, y_1), but again did not find the treasure there. After informing Alice about this, Bob received an answer that another mistake had occurred, and again one letter in s needed to be changed. Bob changed the letter again, repeated the sequence of commands...

This was repeated q times. Bob never found the treasure.

Your task is to identify all the points where Bob tried to find the treasure, in chronological order.

Input

The first line contains string s of length from 1 to 10^5, consisting of letters "E", "N", "S" and "W".

The second line contains an integer q ($1 \le q \le 10^5$).

Each of the following q lines contains an integer i and a letter c, which are space-separated ($1 \le i \le |s|$, $c \in \{$"E", "N", "S", "W"$\}$).

Output

Print $q + 1$ lines containing 2 numbers each — the coordinates of the points where Bob tried to find the treasure, in chronological order. Thus, the first line should contain the coordinates of the point (x_0, y_0) for the initial line s, the following q lines — after each of the q corrections.

Example

standard input	standard output
NEE 4 1 S 1 W 2 E 3 N	2 1 2 -1 1 0 1 0 0 1

Note

In the example, the command strings after each change are the following: "SEE", "WEE", "WEE", "WEN".

Problem F. Fast Food

Input file:	standard input
Output file:	standard output
Time limit:	3 seconds
Memory limit:	128 megabytes

Hard times have come for the network of fast food restaurants Mugdonald's. In order not to go bankrupt, the network management decided to save on labor.

Representatives of the network came to the nearest university and found m hungry students there, ready to work for a penny. Exactly n days left until the end of the working quarter, each student expressed his wishes about the amount of remuneration for work on each of these days. On each working day, the management of the Mugdonald's network must choose exactly one of the students to stand at the checkout. And it is desirable to choose so that in the end to spend as little money as possible on the salaries of employees.

The matter is complicated by the following fact: you have to stand at the cash desk around the clock, and each i-th student cannot stay awake for more than a_i days in a row. Therefore, the schedule must be designed in such a way that workers do not lose consciousness.

Help the Mugdonald's make schedule the way they want.

Input

The first line contains two numbers: n — the number of working days and m — the number of hungry students ($1 \le n \le 1500$, $2 \le m \le 1500$).

The second line contains m integers a_i ($1 \le i \le m$) that describe for each student the maximum number of days in a row that he can stay awake ($1 \le a_i \le n$).

The next m lines contain n non-negative integers each, and the j-th number of the i-th line means the amount of money c_{ij}, for which the i-th student is ready to work in j-th day ($0 \le c_{ij} \le 10^6, 1 \le i \le m, 1 \le j \le n$).

Numbers in lines are separated by spaces.

Output

In the first line print a single number — the minimum amount of money that the management of the Mugdonald's needs to spend.

In the second line print n integers, the i-th of which determines the number of the student who will have to be at the cash desk on the i-th day.

Example

standard input	standard output
5 2 2 2 1 3 6 4 1 5 2 3 1 1	9 1 1 2 2 1

Problem G. Graph Reduction

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	128 megabytes

An undirected graph is called *cubic* if three edges go from each of its nodes. We will consider such graph not containing any loops or multiple edges. It is easy to prove that such graph always contains an even number of nodes.

An undirected graph is called *bipartite* if all its nodes can be divided into two subsets such that there's no edge that would connect two nodes of one and the same subset.

You are given a cubic graph containing $2n$ nodes and $3n$ edges. Remove from it exactly n egdes so that the result was a bipartite graph. Otherwise write that the task is impossible to perform.

Input

The first line contains an integer n ($2 \le n \le 10^5$). Next $3n$ lines describe the graph's edges: the i-th of these lines describes the i-th edge of the graph and contains two integers u and v — the numbers of nodes connected by it ($1 \le u, v \le 2n$).

It is guaranteed that the graph is cubic, that ts exactly three edges go from each node; besides, the graph has no loops and multiple edges.

Output

If there's no solution, then print "`Impossible`" without the quotes.

Otherwise, print "`Possible`" without the quotes; print on the second line space-separated n numbers — the numbers of edges that need to be removed. The edges are numbered starting from one in the order in which they are given in the input data. If there are several solutions, print any of them.

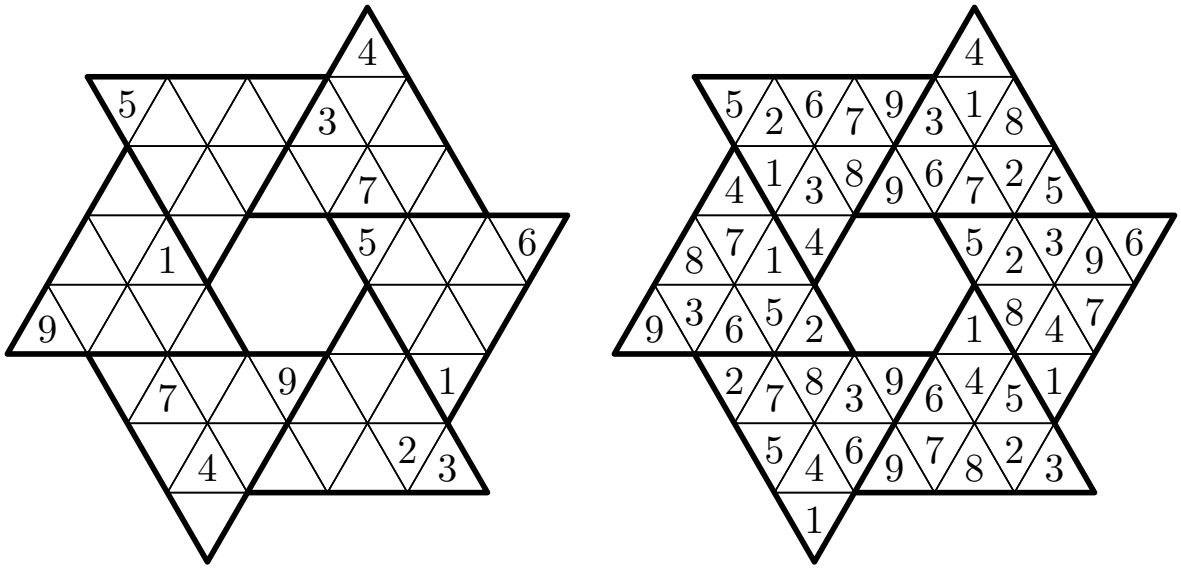
Examples

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

Problem H. Hoshi Sudoku

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	128 megabytes

Hoshi Sudoku is a variation of the popular Sudoku puzzle. The Sudoku grid consists of 6 large triangles divided into 9 small triangles arranged as shown in the figure below. Some triangles contain numbers from 1 to 9.



To solve a Hoshi Sudoku, you need to fill all the small triangles with numbers from 1 to 9 so that in each large triangle, as well as in small triangles on 18 lines (6 lines in each of the 3 directions), each number occurs at most once. Note that 6 lines of 18 contain 8 triangles instead of 9, so only 8 of 9 numbers should be used on these lines, but they must also be pairwise distinct.

A correctly composed Hoshi Sudoku has a unique solution. You need to check if the puzzle is correctly composed, namely, to determine how many solutions the given Hoshi Sudoku has. Two solutions are considered different if there is a triangle in which the numbers in the two solutions differ. Also, if the solution is unique — you need to find that solution.

Input

You are given 8 lines that describe the initial state of the puzzle. In total, there are 54 non-whitespace characters in the input, each of which can be either "`.`", or a number from "`1`" to "`9`". "`.`" means that the corresponding triangle is empty. Triangles are listed line by line from top to bottom, in each line from left to right. See examples for more precise understanding.

It is guaranteed that in the input any of the 6 large triangles and any of the 18 lines contains each number from 1 to 9 at most once.

Output

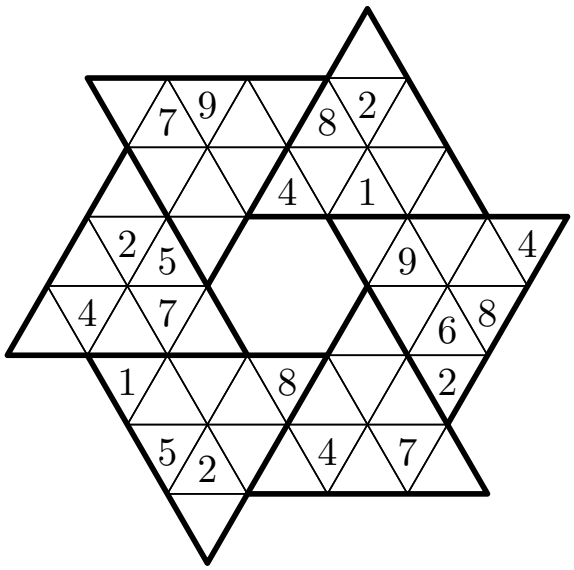
Print "`No solutions`" if the puzzle has no solutions. If there is only one solution — print "`Single solution`" in the first line, and in the next 8 lines print the solution itself in the same format as in the input. If there is more than one solution — print "x `solutions`", where x is the number of solutions.

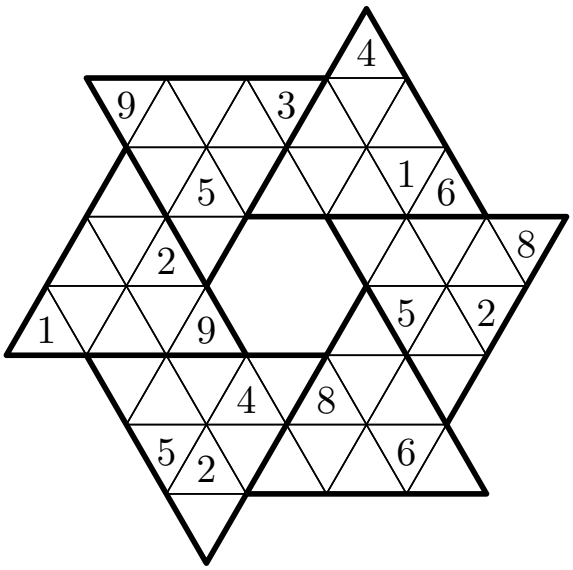
Examples

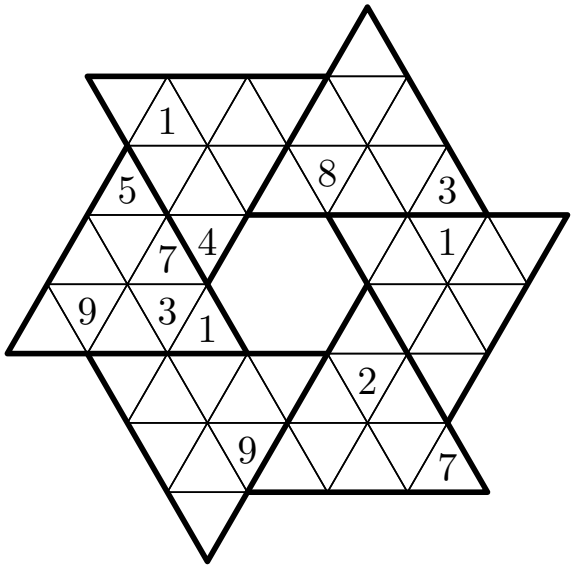
standard input	standard output
4 5....3.. ......7.. ..1. 5...6 9.... .... .7..9...1 .4....23 .	Single solution 4 52679318 413896725 8714 52396 93652 1847 278396451 54697823 1
4 5....3.. ......7.. ..1. 5...6 9.... .... .7..9...1 .4..8.23 .	No solutions
4 .....3.. ......7.. ..1. ....6 9.... .... .7..9.... .4.....3 .	456 solutions
. ...	801655626240 solutions

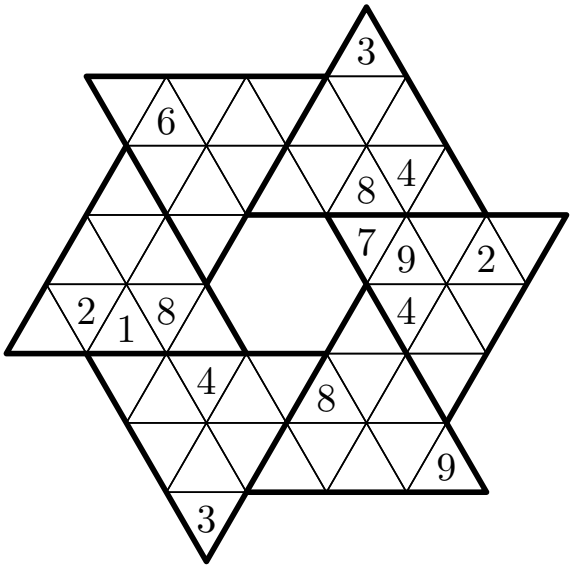
Note

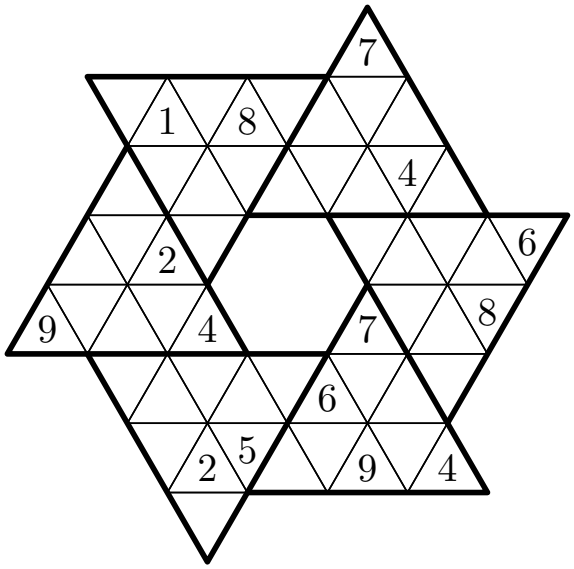
As a bonus, here are some more Hoshi Sudokus. You can use them to test the program, or just try to solve by hands after the contest. Each of these puzzles has a unique solution.

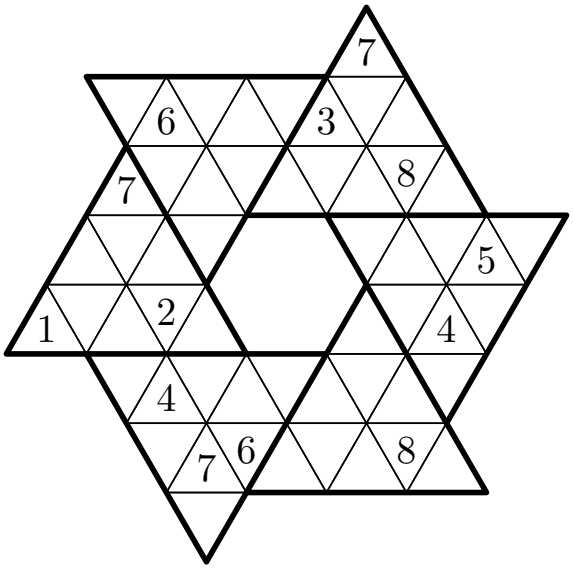

7 9 8 2
4 1
2 5 9 4
4 7 6 8
1 8 2
5 2 4 7


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


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


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


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


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

Problem I. Imbalance

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	128 megabytes

A rooted tree is called *binary* if each node has no more than two children connected to it a level below from the left and from the right. Thus, any node of a binary tree can be reached from the root by sequentially moving down to the left or right child node. These child nodes can also be considered as the roots of subtrees placed below the given nodes. We will refer to these subtrees as the left and right subtrees. If there is no child node on the left or right — we will assume that the corresponding subtree is *empty*.

The *height* of a binary tree is the length of the longest path from the root to the node below it in the tree. We assume that the height of an empty subtree is zero. A binary tree is called *balanced* if, for any node, the heights of the left and right subtrees differ by no more than one. A binary tree is called *perfectly balanced* if, for any node, the heights of the left and right subtrees are equal.

It is clear that there are balanced binary trees of varying degrees of perfection. Let the *imbalance* of a balanced binary tree be the number of nodes for which the heights of the left and right subtrees differ by exactly one.

Count the number of balanced binary trees of n vertices that have imbalance k. Two trees are considered different if there is a sequence of movings down left-right from the root such that in one tree it leads to some node, but not in the other. Since the answer can be very large, calculate it modulo $10^9 + 7$.

Input

You are given two integers n and k ($1 \le n \le 10^5$, $0 \le k \le 20$).

Output

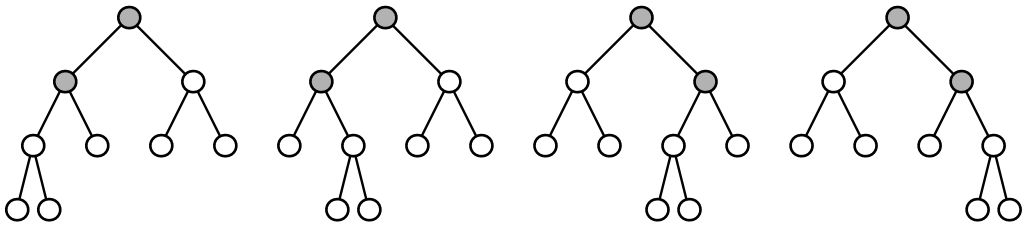
Print one number — the answer modulo $10^9 + 7$.

Examples

standard input	standard output
7 0	1
9 2	4
20 4	480

Note

The figure below shows all 4 variants of balanced binary trees with 9 nodes and imbalance 2. Nodes that give the imbalance are marked in gray.



Problem J. Jurassic Park

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	128 megabytes

Today Maxim visited the Jurassic Park, where he saw many models of dinosaurs. Some of them were life-sized now-extinct dinosaurs, and some even moved.

Maxim wrote down the names of the dinosaurs he liked most in his notebook in order to search for additional information on them at home in the Internet. Arriving home, Maxim found that the list turned out to be too large in order to study all these dinosaurs in one day. So he decided to split the study into 3 days as follows:

- Dinosaurs whose names end with "`saurus`" will be studied by Maxim on the first day.
- Dinosaurs whose names end in "`raptor`", "`ceratops`", "`odon`", "`pteryx`", or "`mimus`", Maxim will study on the second day.
- On the third day, Maxim will study all the other dinosaurs.

From the list of dinosaur names, determine how many dinosaurs Maxim will study on each of the 3 days.

Input

The first line contains an integer n — the number of dinosaur names in the list ($1 \le n \le 100$). Next come n names of dinosaurs, one per line. Each name is a string consisting of small Latin letters from 5 to 23 characters long. It is guaranteed that all dinosaur names in the list are pairwise distinct.

Output

Print 3 numbers, separating them with a space — the number of dinosaurs Maxim will study on the first, second and third days respectively.

Example

standard input	standard output
12 tyrannosaurus micropachycephalosaurus velociraptor triceratops diplodocus graciliceratops stenopelix piatnitzkysaurus iguanodon brachiosaurus caudipteryx harpymimus	4 6 2

Problem K. Katana vs. Cake

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	128 megabytes

Today is Samurai Jack's birthday. Jack's friends decided to arrange a surprise and brought a huge cake in the form of a ball. But bad luck — the cake is so huge that it is not clear how to cut it, and, most importantly, with what.

Without thinking twice, Samurai Jack pulled out his katana and in a split second the cake was cut into several pieces. In total, Jack made n cuts, and his movements were so fast and precise, that none of the pieces moved in the process of cutting. Thus, if we consider the cake to be an ideal ball of radius $R = 1$, then as a result of cutting the ball turned out to be cut by n planes.

Who will get the biggest piece? Of course, samurai Jack, as a birthday man.

Determine the volume of a piece of cake that Jack will get.

Input

The first line contains an integer n — the number of cutting planes ($1 \le n \le 5$).

Each of the following n lines contains 4 real numbers A, B, C, and D with a precision of one decimal place, which define the equations of cutting planes of the form $Ax + By + Cz + D = 0$ ($-3 \le A, B, C, D \le 3$).

It is guaranteed that for all equations of the planes A, B and C are not simultaneously equal to zero. No two planes match. It is also guaranteed that each plane passes at a distance of at most 0.9 from the center of the ball. The center of the ball is at the point with coordinates $(0, 0, 0)$.

Output

Print one real number — the volume of the largest part of the ball of radius $R = 1$ after it has been cut by n planes. Your answer will be accepted if it differs from the jury's answer by no more than 10^{-4} in relative or absolute value.

Example

standard input	standard output
3 1.0 2.0 1.0 0.7 -0.6 0.0 1.0 0.2 1.0 -3.0 1.0 0.5	1.051397

Problem L. Lusine's Sausages

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	128 megabytes

Aunt Lusine has n pigs at the pig farm. They can be numbered by consecutive integers from 1 to n. The i-th pig can produce a_i kilograms of meat and b_i kilograms of fat.

Aunt Lusine recently received an order for k kilograms of sausages. To fulfill the order, aunt Lusine is going to put a subset of her pigs under the knife. All the meat and fat obtained as a result will be used for the production of sausages, nothing will be thrown away or added. So, if we fix a subset of pigs S, then from them we can get $W(S) = \sum_{i \in S}(a_i + b_i)$ kilograms of sausages. S should be such that $W(S) \geq k$ is fulfilled (aunt Lusine plans to give surplus sausages to her nephews).

The high content of meat in sausages is a sign of high quality. Aunt Lusine wants to make sausages of the highest quality. Thus, the amount of meat for a subset of pigs S is defined as $M(S) = \sum_{i \in S} a_i$, and the percentage of meat in sausages is defined as $Q(S) = M(S)/W(S) \times 100\%$.

It is considered that sausages are of *outstanding* quality, if the percentage of meat in them is 70% or more. *Good* quality sausages are obtained with a meat content of at least 50%. If the percentage is less than 50% — there are sausages of *low* quality. Determine what the highest quality of sausages can be achieved.

Input

The first line contains two integers n and k ($1 \leq n \leq 100$, $1 \leq k \leq 20000$).

The second line contains n integers, the i-th of them is equal to a_i ($10 \leq a_i \leq 100$, $1 \leq i \leq n$).

The third line contains n more integers — the values of b_i ($10 \leq b_i \leq 100$, $1 \leq i \leq n$).

Output

Print one of the lines:

- "`Outstanding`", if it is possible get outstanding quality sausages.
- "`Good Quality`", if only good quality sausages can be achieved.
- "`Low Quality`", if only low quality can be achieved.
- "`Reject`", if the order cannot be fulfilled in principle and aunt Lusine should refuse it.

Examples

standard input	standard output
3 100 70 50 20 30 50 80	Outstanding
3 150 70 50 20 30 50 80	Good Quality
3 201 70 50 20 30 50 80	Low Quality
3 301 70 50 20 30 50 80	Reject

Problem M. Monetary Reform

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	128 megabytes

Once in Berland, the government decided to abandon the old money and replace it with new ones. Thus, the Central Bank of Berland has already launched the printing of new n banknotes of various denominations.

Denominations of new banknotes will be denoted as c_1, c_2, and so on up to c_n burles. The order of printing new money has already been approved and is carried out in the following order: first banknotes with denomination c_1 are printed and put into circulation, then banknotes with denomination c_2 are printed, and as soon as they are put into circulation, banknotes with denomination c_3 are printed, and so on.

When the process was already underway, the Berland government was horrified to discover that the banknotes with denomination 1 would be the last to be put into print. That is, $c_n = 1$. And until that time, some amounts of money cannot be collected with banknotes of existing denominations without change.

In order to estimate the scale of the consequences of the error, the government of Berland wants to determine the maximum amount that cannot be collected with the available banknotes after each of n printing iterations.

Input

The first line contains an integer n ($1 \le n \le 100$).

The i-th of the following lines contains an integer c_i ($2 \le c_i \le 40000$ for $1 \le i < n$, $c_n = 1$).

It is guaranteed that all values of c_i are pairwise distinct.

Output

Print n lines. In the i-th of them print the maximum amount that cannot be made with denominations from c_1 to c_i inclusive. If the maximum amount is not limited from above — print "`INF`" instead of a number. See example for a clearer understanding.

Example

standard input	standard output
4 10 3 5 1	INF 17 7 -1

Note

In the example, after entering $c_1 = 10$ into circulation, it is impossible to collect any amount that is not divisible by 10.

After entering $c_2 = 3$, it is possible to get 3, $3 + 3 = 6$, $3 + 3 + 3 = 9$, 10, $3 + 3 + 3 + 3 = 12$, $10 + 3 = 13$, $3 + 3 + 3 + 3 + 3 = 15$, $10 + 3 + 3 = 16$, $3 + 3 + 3 + 3 + 3 + 3 = 18$, $10 + 3 + 3 + 3 = 19$, $10 + 10 = 20$. Further, by adding the required number of banknotes with denomination 3 to the amounts of 18, 19 and 20, it is possible to collect any amount. So 17 is the maximum amount that cannot be collected.

Adding the denomination $c_3 = 5$, we get the amounts 3, 5, $3 + 3 = 6$, $5 + 3 = 8$, $3 + 3 + 3 = 9$, 10.

With the addition of $c_4 = 1$, we can collect any non-negative amount.