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 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 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 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 N. Needle Navigator

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	128 mebibytes

The API of the compass simulator is next: when the compass needle is currently in some direction (between 0 and 359 degrees, with north being 0, east being 90), it is being animated by giving the degrees to spin it.

For example, if the needle is pointing north, and you give the compass an input of 90, it will spin clockwise (positive numbers mean clockwise direction) to stop at east, whereas an input of -45 would spin it counterclockwise to stop at north west.

Your task is to find the *shortest path* (angle) from the current needle direction to the correct direction.

Input

The first line of input contains an integer n_1 ($0 \leq n_1 \leq 359$), the current direction of the needle. The second line of input contains an integer n_2 ($0 \leq n_2 \leq 359$), the correct direction of the needle.

Output

Output the change in direction that would make the needle spin the shortest distance from n_1 to n_2. A positive change indicates spinning the needle clockwise, and a negative change indicates spinning the needle counter-clockwise. If the two input numbers are diametrically opposed, the needle should travel clockwise.

Example

standard input	standard output
315 45	90
180 270	90
45 270	-135

Problem O. Obstacles And Stones

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	128 mebibytes

Consider a 2D grid, which contains stones, obstacles, and open spaces. Gravity will pull the stones straight down, until they hit an obstacle, or the bottom of the grid, or another stone which has already come to rest. Obstacles don't move. Given such a grid, determine where the stones eventually settle.

Input

The first line of input contains two integers, r and c ($1 \leq r, c \leq 100$), which are the number of rows and the number of columns of the grid. On each of the next r lines will be c characters: '`o`' for a stone, '`x`' for an obstacle, and '`.` for an open space.

Output

Output the grid, after the stones have fallen.

Examples

standard input	standard output
4 4 oooo x... ..x. x.xx	o... x.o. ..xo xoxx
4 3 o.o o.o o.. ...	... o.. o.o o.o

Problem P. Powerful Attack

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	128 mebibytes

In the game ¡¡Heroes of the Fight and Rage¿¿ you are fighting dangerous monster. The monster has the following moves:

- Rake, denoted by the letter '`R`';
- Bite, denoted by the letter '`B`';
- Laser breath, denoted by the letter '`L`'.

In order to defend, you must perform a counter move per move that the monster makes:

- Slice, denoted by the letter '`S`', counters the monster's rake;
- Kick, denoted by the letter '`K`', counters the monster's bite;
- Shield, denoted by the letter '`H`', counters the monster's laser breath;

However, there is one catch. When the monster performs a subsequent combination of the three moves Rake, Bite and Laser breath, in any order, it becomes a very powerful attack for which you must perform a single counter move called Combo breaker, denoted by the letter '`C`'. A single Combo breaker absorbs the entire combination of three moves. Any following moves from the monster will have to be countered separately or as part of a new combination. A move of the monster can never be part of more than one combination.

Given a string, representing the monster moves, print the sequence of your actions to counter all monster's attacks.

Input

A single line containing a string of at least 1 and at most 10^6 characters, consisting of the letters '`R`', '`B`' and '`L`'.

Output

Output a single string consisting of the letters denoting the moves that are to be made in succession in order to defend from the monster's attacks.

Example

standard input	standard output
RRBBBLLR	SSKKKHHS
RBLLLBRR	CHCS
RBLBR	CKS

Problem Q. Quickest Way To Earn Money

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	128 mebibytes

Foretelling the future is hard, but imagine if you could just go back in time and use your knowledge of stock price history in order to maximize your profits!

Lets you can go back in time a few days and bring a measly 100 dollars with you. How much money could you make by just buying and selling stock of one company at the right times?

Note that you can not buy fractional shares, you must buy whole shares. The total number of shares is 100 000, so you can not own more than 100 000 shares at any time.

For simplicity, there are no fees for buying and selling stocks, stock prices change only once per day, and your trading does not influence the valuation of the stock.

Input

The first line of input contains an integer d ($1 \leq d \leq 365$), the number of days that you can go back in time. Then follow d lines, the i'th of which contains an integer p_i ($1 \leq p_i \leq 500$) giving the price at which you can buy or sell stock on day i. Days are ordered from oldest to newest.

Output

Output the maximum possible amount of money you can have on the last day. Note that the answer may exceed 2^{32}.

Example

standard input	standard output
6 100 200 100 150 125 300	650

Problem R. Regular Strings

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	128 mebibytes

Define a *k-regular* string as follows:

A string s is k-regular if the length of the string $|s|$ is a multiple of k, and if you chop the string up into $|s|/k$ substrings of length k, then each of those substrings (except the first) is the same as the previous substring, but with its last character moved to the front.

For example, the following string is 3-regular: "`xyzzxyyzxxyz`" The above string can break up into substrings "`xyz`", "`zxy`", "`yzx`", and "`xyz`", and each substring (except the first) is a right-rotation of the previous substring.

Given a string, determine the smallest k for which the string is k-regular.

Input

The single line of input contains a string s ($1 \leq |s| \leq 100$) consisting only of lowercase English letters.

Output

Output the integer k, which is the smallest k for which the input string is k-regilar

Examples

standard input	standard output
qqqqqq	1
cddccddccddc	2