Problem A. Amidakuji

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

Ghost Leg, known in Japan as Amidakuji, is a method of lottery designed to create random pairings between two sets of any number of things, as long as the number of elements in each set is the same. This is often used to distribute prizes among people, where the number of prizes distributed is the same as the number of people.

It consists of N vertical segments numbered from 1 to N with horizontal segments connecting two adjacent vertical segments scattered randomly along their length; no two horizontal segments (also called "legs") have the same height, so they can be numbered by integers from 1 to M. i-th leg connects two vertical lines p_i and $p_i + 1$ $(1 \leq p_i \leq N - 1)$. The number of vertical lines equals the number of people playing, and at the bottom of each vertical line there is an prize that will be paired with a player. The general rule for playing this game is: choose a line on the top, and follow this line downwards. When a horizontal line is encountered, follow it to get to another vertical line and continue downwards. Repeat this procedure until reaching the end of the vertical line. Then the player is given the prize placed at the bottom of the line.

You have an Amidakuji board and your task is to process queries of two type:

- `A` i — remove i-th (in initial numeration) leg. Note that after removal legs are **not** renumbered.
- `B` j — check number of the prize for j-th player to win with current board (i.e. 1-based number of the vertical line where the player ends up at the bottom).

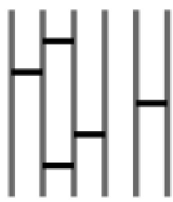
Input

First line of the input contains two integrs N and M $(2 \leq n \leq 10^6, 1 \leq m \leq 10^6)$ — number of vertical and horizontal lines, respectively. Next line contains M integers p_i $(1 \leq p_i \leq N - 1)$ — number of leftmost of two vertical lines connected by the respective horizontal line. Next line contains one integer Q $(1 \leq Q \leq 10^6)$ — number of queries. Then Q lines follow, describing the queries: '`A`' then integer i $(1 \leq i \leq M)$ for removal of the leg and or '`B`' then integer j $(1 \leq j \leq N)$ to query number of prize for ij-th player. You may assume that no leg is removed twice and that atleast one query have type B.

Output

For each query of type B print one integer — an answer to this query.

Example

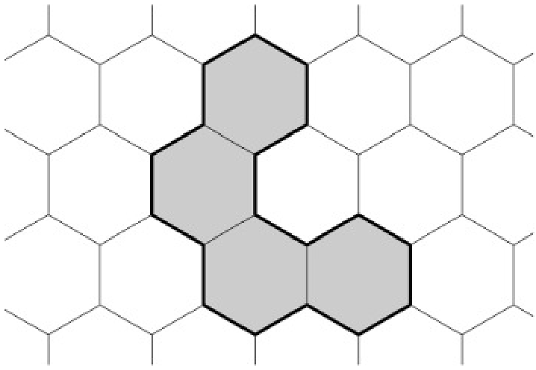
standard input	standard output	Notes
6 5 2 1 5 3 2 10 B 1 B 2 B 6 A 3 B 6 B 3 A 1 B 2 B 3 B 4	3 4 5 6 1 1 4 2	 1 2 3 4 5 6

Problem B. Bees

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

Byteasar is a beekeeper. His hobby is collecting different patterns in which bees put honey into cells. Each pattern is a connected set of hexagonal cells without holes. (More formally, a pattern is a subset of a hexagonal grid of edge length 1. Every two hexagons that belong to the pattern are connected by a path of edge-adjacent hexagons that belong to the pattern. Moreover, every two grid hexagons that do not belong to the pattern are connected by a path of edge-adjacent hexagons that do not belong to the pattern.)

Byteasar wants to make a comprehensive list of patterns of different features — number of cells and perimeter. You could help him by writing a program that, given two numbers n and p, checks whether there is a pattern consisting of n cells with perimeter equal to p. For example, the following picture shows a pattern with $n = 4$ and $p = 18$.



Input

The first line of input contains one integer t ($1 \leq t \leq 1000$), the number of test cases. Each of the following t lines contains two integers n and p ($1 \leq n, p \leq 1000$) which specify the number of hexagons and the perimeter of a pattern.

Output

Your program should output t lines with answers to the respective test cases. If there is no pattern with the given features, output a single word "`NO`". Otherwise output a string of length p consisting of letters "`L`" and "`R`". The string should describe a walk around the pattern's perimeter (in clockwise or counterclockwise direction). The letter "`L`" denotes a left turn, and the letter "`R`" denotes a right turn.

If there is more than one pattern, print any one of them.

Examples

standard input	standard output
3 4 18 3 18 3 12	RLRRRRLRLRLRRRRLRL NO LRRRLRRRLRRR

Problem C. Cutting

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

You are given string S and list M, consisting of N words. During an operation you may choose substring of string S, if it can be found as word in list M, and cut it out of string S. Then remaining parts of string S are merged if there are any. Determine minimal amount of opertions required to erase whole string S. It is guaranteed that it can be done.

Input

The first line contains word S ($1 \le |S| \le 100$). Second line contains integer N, amount of words in the list ($1 \le N \le 100$). It is followed by N lines listing words from M. All words M_i consist of lowercase latin letters only, $1 \le |M_i| \le 100$.

Output

Print one integer, that is minimal number of opertions required to erase whole string S.

Example

standard input	standard output
abacaba 4 aba aca a b	3

Note

First operation cuts substring "`aca`", result "`abba`".

Second operation cuts substring "`b`", result "`aba`".

Third operation erases string aba.

Problem D. Duel

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2.5 seconds
Memory limit:	256 mebibytes

Two players play a game. Each of players has initial amount of cards (n_1 and n_2 correspondently). On every turn players choose one card each and open it. Weaker card goes to retreat, the stronger one is taken back by the player who opened it. If the players have shown same cards, both go to retreat. The game is continued, until at least one of the players is over of cards. If one of the players still has at least a card when that happens, than he gets 1 point, and his rival 0. If both of the players are over of cards, than each gets 0.5 points. There are N kinds of cards totally. Strength relation of cards is nontransitive and defined with matrix A. If card i is stronger than j, A_{ij} is 1, and 0 otherwise. Define price of that game for the first player, supposing that second player plays optimally.

Input

The first line contains an integer N ($1 \le N \le 8$). Following N lines contain N numbers each, which define matrix A. ($A_{ij} \in \{0, 1\}$, $A_{ij} + A_{ji} = 1$ for any two $i \ne j$, $A_{ii} = 0$.

Next line contains integer n_1 ($1 \le n_1 \le 8$), then n_1 integers follow, each descripting a kind of corresponding card of the first player. Last line contains integer n_2 ($1 \le n_2 \le 8$), then n_2 integers follow, each descripting a kind of corresponding card of the first player.

Output

Output price of the game for the first player with accuracy no less than 10^{-8}.

Examples

standard input	standard output
3 0 1 1 0 0 1 0 0 0 2 3 2 1 1	0.00000000
3 0 1 0 0 0 1 1 0 0 3 1 2 3 3 1 2 3	0.50000000
3 0 1 0 0 0 1 1 0 0 3 1 2 3 3 2 2 3	0.66666667

Problem E. Elimination

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2.5 seconds
Memory limit:	256 mebibytes

There are N points on a plane. Each of them is colored in one of two colors: black and white. Points are involved in such process: on each iteration any point which sees at least one point of another color is eliminated. Point A see point B, if the segment AB doesn't contain other non-eliminated points.

During one iteration, all points which must be eliminated at this iteration are eliminated simultaneously. If there are no more points of any color, the process ends.

Determine how many iterations it requires to finish the process, and what color will survive if any.

Input

The first line contains number N ($1 \le N \le 100\,000$) — number of points on the plane. Next N lines contain three integers each: x_i, y_i and $color_i$ ($0 \le x_i, y_i \le 1\,000\,000$, $color_i \in \{0, 1\}$) — coordinates and color of i-th point. 0 means white, 1 means black. No two points have the same coordinates.

Output

Output the word "`Draw`", if all points will be eliminated, "`Black`", if black points will survive, or "`White`" if white will survive. In the same line output number of iterations (separated by space).

Example

standard input	standard output
3 0 0 0 1 0 1 0 1 1	Draw 1
3 0 0 0 1 0 1 2 0 1	Black 1

Problem F. Far Far Away Point

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

You are given parallelepiped with length A, width B and height C. Find surface distance to the most distant (by surface) point from vertex of parallelepiped.

Input

The only line has three integers A, B, C ($1 \le A, B, C \le 1000$).

Output

The only real number — a distance with absolute or relative error 10^{-8} or better.

Example

standard input	standard output
`1 1 1`	`2.2360679774998`

Problem G. Grouping

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

You are given N integers a_i. Let b_i are K numbers (not necessarily integer), such as :

$$S = \sum_{i=1}^{N} \min_{i \le j \le K} |a_i - b_j|$$

is minimal. Your task is to calcilate S.

Input

First line of the input contains two integers N and K ($1 \le N \le 5000$, $1 \le K \le N$). Second line of the input contains N integers a_i ($0 \le a_i \le 4 \cdot 10^4$).

Output

Print real number S with absolute or relative error not greater than 10^{-8}.

Example

standard input	standard output
5 3 1 5 7 10 14	5

Note

The b_i for the sample may look as $\{1, 7, 14\}$.

Problem H. Hidden Triangles

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

N triangles are put on the plane in order from 1 to N. Each triangle is opaque and closes everything behind it.

Determine, which of the triangles are visible on the plane, i.e. triangles with atleast one part of positive area, which is not covered by any other triangle.

Input

First line contains one integer N ($1 \le N \le 500$) — amount of triangles. Each of next N lines describe one triangle; triangles are listed in the order they were put to the plane. Each triangle is defined by 6 integers x_1, y_1, x_2, y_2, x_3, y_3 — coordinates of vertices ($-1000 \le x_i, y_i \le 1000$). All the triangles are not degenerated. Any edge of a triangle has no more than one common point with any edge of anoter triangle.

Output

Print number of visilbe triangles in the first line. In second line list those triangles in arbitrary order.

Example

standard input	standard output
3 1 0 4 0 0 3 -2 1 5 -2 3 4 -2 2 4 1 2 4	2 2 3

Problem I. Irreducible Polynomials

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

There is prime number p. $Z_p = \{0, 1, ..., p-1\}$ set of integers modulo p. At this field multiplication and addition operations are done modulo p. Now consider irreducible monic polynomials on this field of a kind:

$$f(x) = x^n + a_{n-1}x^{n-1} + \ldots + a_1x + a_0$$

where n is power of the polynominal, x is variable, $a_i \in Z_p$ are coefficients. Value x which makes $f(x)$ equal to zero is a root of this polynomial. Let's expand the set, so any polynomial of any power, which is no less than one, has at least one root.

Monic polynomial is irreducible, if it has a root which is not a root of any monic polynomial of lower power with coefficients from Z_p. For example, polynomial $x^2 + x + 1$ in the field Z_2 is irreducible. Its root (labeled as e_2) is not a root of polynomial x, nor $x + 1$. Same can be said about second root $1 + e_2$ of specified polynomial. And this is only irreducible polynomial in the set Z_2.

Your task is to determine amount of monic irreducible polynomials of power n in the set Z_p. Because this number can be large, it's required to get the remainder of division of that number by m.

Input

The input contains three integers p, n, m ($1 \le p, n, m \le 10^9$), p is prime.

Output

Output number of irreducible polynomials of power n in the field Z_p modulo m.

Example

standard input	standard output
2 2 10	1
3 4 100	18

Problem J. John and David Blaine

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

Once upon a time John was preparing for an international math olympiad and fell asleep. John was a huge fan of all magicians, and in his sleep, he saw David Blaine who said:

"Wanna see some math magic? Look. I write N positive integers greater than 1 and less than 2^{63} in a row on the blackboard. Some of them are equal: at least one of the integers is equal to some other integer in that row. Below I write $K < N$ positive integers greater than 1 and less than 2^{63}, some of the integers in the new row are also equal. Now note that the product of all integers in the lower row is equal to one plus the product of all integers in the upper row.

And now, now I remove some of the duplicates in the upper row. That is, the set of integers which occur at least once in the upper row remains the same, but I erase at least one element of the row. And then I remove some of the repetitions from the lower row as well. Once again, no integer is completely gone from the lower row, I just erase at least one repetition. Now, let us calculate the products again... As if nothing ever happened, the product of all integers in the lower row is equal to one plus the product of all integers in the upper row. Magic! Boom!"

And then John woke up. He was extremely excited about the trick, but suddenly understood that the only number he remembers is N. Help John to repeat the trick!

Input

The first line of input holds a single integer N ($2 \le N \le 100$), the number of integers in the upper row written by David Blaine.

Output

On the first line, output the upper row of integers separated by spaces. On the second line, output the lower row. On the third and fourth lines, output the upper and lower rows, respectively, after some repetitions were removed from them. All integers must be between 2 and $2^{63} - 1$, inclusive. Each row must be sorted in non-decreasing order.

If multiple answers are possible, output any one of them. If it is impossible to do the trick with N numbers, output a single word "`Impossible`" on the first line instead.

Examples

standard input	standard output
`2`	`Impossible`

`3`	`2 2 2`
	`3 3`
	`2`
	`3`