TASK	PET	KEMIJA	CROSS	MATRICA	BST	NAJKRACI
input	standard input					
output	standard output					
time limit	1 second	1 second	1 second	0.2 seconds	1 second	5 seconds
memory limit	32 MB	32 MB	32 MB	32 MB	32 MB	32 MB
points	**30**	**40**	**70**	**110**	**120**	**130**
	500					

In the popular show "Dinner for Five", five contestants compete in preparing culinary delights. Every evening one of them makes dinner and each of other four then grades it on a scale from 1 to 5.

The number of points a contestant gets is equal to the sum of grades they got. The winner of the show is of course the contestant that gets the most points.

Write a program that determines the winner and how many points they got.

INPUT

Five lines, each containing 4 integers, the grades a contestant got.

The contestants are numbered 1 to 5 in the order in which their grades were given.

The input data will guarantee that the solution is unique.

OUTPUT

Output on a single line the winner's number and their points, separated by a space.

EXAMPLES

input	input
5 4 4 5 5 4 4 4 5 5 4 4 5 5 5 4 4 4 4 5	4 4 3 3 5 4 3 5 5 5 2 4 5 5 5 1 4 4 4 4
output	**output**
4 19	2 17

Luka is fooling around in chemistry class again! Instead of balancing equations he is writing coded sentences on a piece of paper. Luka modifies every word in a sentence by adding, after each vowel (letters 'a', 'e', 'i', 'o' and 'u'), the letter 'p' and then that same vowel again.

For example, the word "kemija" becomes "kepemipijapa" and the word "paprika" becomes "papapripikapa". The teacher took Luka's paper with the coded sentences and wants to decode them.

Write a program that decodes Luka's sentence.

INPUT

The coded sentence will be given on a single line. The sentence consists only of lowercase letters of the English alphabet and spaces. The words will be separated by exactly one space and there will be no leading or trailing spaces. The total number of character will be at most 100.

OUTPUT

Output the decoded sentence on a single line.

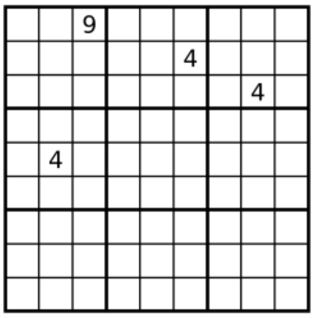
EXAMPLES

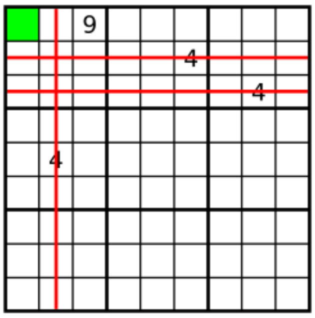
input
`zepelepenapa papapripikapa`
output
`zelena paprika`
	input
	`bapas jepe doposapadnapa opovapa kepemipijapa`
	output
	`bas je dosadna ova kemija`

In the game of Sudoku, the objective is to place integers between 1 and 9 (inclusive) into a 9x9 grid so that each row, each column, and each of the nine 3x3 boxes contains all nine numbers. The starting board is partially filled in so that it is possible to logically deduce the values of the other cells. Sudoku puzzles range in difficulty, and complex analysis methods are required to solve the hardest puzzles. In this problem, however, you will implement one of the simplest methods, cross-hatching.

In cross-hatching, we select one of the nine numbers and, for each of its occurrences in the grid, cross out the corresponding row, column and 3x3 box. Now look for any 3x3 boxes where there is only one possible placement for the number and place it there.

The first image below shows a very sparsely filled in Sudoku grid. However, even in this grid it is possible to deduce using cross-hatching that the number in the top left cell is 4, as illustrated in the second image.





You will be given a partially filled-in grid. Your task is to repeatedly apply the cross-hatching method for different numbers until no more deductions can be made about any number.

The initial placement of the numbers in the grid may be invalid. It is also possible that there will be no available cell for a number in a 3x3 box. In both cases, you are to report an error.

INPUT

Input will consist of 9 lines, each containing exactly 9 characters. Each character will either be a digit between 1 and 9, or a period ('.') denoting an empty cell.

OUTPUT

If the input is valid and there is no contradiction while solving, you should output the grid in the same format it was given in, with cells filled in if their value can be deduced using cross-hatching. Otherwise, output "ERROR" (quotes for clarity).

EXAMPLES

input ..9...... .....4... ....4. .... .4....... .... **output** 4.9...... .....4... ....4. .... .4.......	**input** ...1...6. 18...9... ..7.642.. 2.9..6.5. .43...72. .6.3..9.1 ..265.1.. ...2...97 .5...3... **output** 524137869 186529473 397864215 219476358 843915726 765382941 972658134 638241597 451793682	**input** 1........ ..1...... ....1. .... **output** ERROR	**input** ....2 ....1.... 1........ ......1.. ....1. .... **output** ERROR

A matrix is a rectangular table of letters. A square matrix is a matrix with an equal number of rows and columns. A square matrix M is called **symmetric** if its letters are symmetric with respect to the main diagonal ($M_{ij} = M_{ji}$ for all pairs of i and j).

The following figure shows two symmetric matrices and one which is not symmetric:

`AAB`
`ACC`
`BCC`
	`AAA`
	`ABA`
	`AAA`
		`ABCD`
		`ABCD`
		`ABCD`
		`ABCD`
			`AAB`
			`ACA`
			`DAA`

Two symmetric matrices.		Two matrices that are not symmetric.	

Given a collection of available letters, you are to output a **subset of columns** in the **lexicographically smallest symmetric** matrix which can be composed using **all** the letters.

If no such matrix exists, output "IMPOSSIBLE".

To determine if matrix A is lexicographically smaller than matrix B, consider their elements in row-major order (as if you concatenated all rows to form a long string). If the first element in which the matrices differ is smaller in A, then A is lexicographically smaller than B.

INPUT

The first line of input contains two integers N ($1 \le N \le 30000$) and K ($1 \le K \le 26$). N is the dimension of the matrix, while K is the number of distinct letters that will appear.

Each of the following K lines contains an uppercase letter and a positive integer, separated by a space. The integer denotes how many corresponding letters are to be used. For example, if a line says "A 3", then the letter A must appear three times in the output matrix.

The total number of letters will be exactly N^2. No letter will appear more than once in the input.

The next line contains an integer P ($1 \le P \le 50$), the number of columns that must be output.

The last line contains P integers, the indices of columns that must be output. The indices will be between 1 and N inclusive, given in increasing order and without duplicates.

OUTPUT

If it is possible to compose a symmetric matrix from the given collection of letters, output the required columns on N lines, each containing P character, without spaces. Otherwise, output "IMPOSSIBLE" (quotes for clarity).

SCORING

In test cases worth 60% of points, N will be at most 300.

In test cases worth 80% of points, N will be at most 3000.

EXAMPLES

input 3 3 A 3 B 2 C 4 3 1 2 3 **output** AAB ACC BCC	**input** 4 4 A 4 B 4 C 4 D 4 4 1 2 3 4 **output** AABB AACC BCDD BCDD	**input** 4 5 E 4 A 3 B 3 C 3 D 3 2 2 4 **output** AC BE DE ED	**input** 4 6 F 1 E 3 A 3 B 3 C 3 D 3 4 1 2 3 4 **output** IMPOSSIBLE

A binary search tree is a tree in which every node has **at most** two children nodes (a left and a right child). Each node has an integer written inside it. If the number X is written inside a node, then the numbers in its left subtree are less than X and the numbers in its right subtree are greater than X.

You will be given a sequence of integers between 1 and N (inclusive) such that each number appears in the sequence exactly once. You are to create a binary search tree from the sequence, putting the first number in the root node and inserting every other number in order. In other words, run insert(X, root) for every other number:

```
insert( number X, node N )
    increase the counter C by 1
    if X is less than the number in node N
        if N has no left child
            create a new node with the number X and set it to be the left child of node N
        else
            insert(X, left child of node N)
    else (X is greater than the number in node N)
        if N has no right child
            create a new node with the number X and set it to be the right child of node N
        else
            insert(X, right child of node N)
```

Write a program that calculates the value of the counter C after every number is inserted. The counter is initially 0.

INPUT

The first line contains the integer N ($1 \le N \le 300000$), the length of the sequence.

The remaining N lines contain the numbers in the sequence, integers in the interval [1, N]. The numbers will be distinct.

OUTPUT

Output N integers each on its own line, the values of the counter C after each number is inserted into the tree.

SCORING

In test cases worth 50% of points, N will be at most 1000.

EXAMPLES

input 4 1 2 3 4 **output** 0 1 3 6	**input** 5 3 2 4 1 5 **output** 0 1 2 4 6	**input** 8 3 5 1 6 8 7 2 4 **output** 0 1 2 4 7 11 13 15

A road network in a country consists of N cities and M one-way roads. The cities are numbered 1 through N. For each road we know the origin and destination cities, as well as its length.

We say that the road F is a **continuation** of road E if the destination city of road E is the same as the origin city of road F. A **path** from city A to city B is a sequence of road such that origin of the first road is city A, each other road is a continuation of the one before it, and the destination of the last road is city B. The length of the path is the sum of lengths of all roads in it.

A path from A to B is a **shortest** path if there is no other path from A to B that is shorter in length.

Your task is to, for each road, output **how many different** shortest paths containing that road, modulo 1 000 000 007.

INPUT

The first line contains two integers N and M ($1 \leq N \leq 1500$, $1 \leq M \leq 5000$), the number of cities and roads.

Each of the following M lines contains three positive integers O, D and L. These represent a one-way road from city O to city D of length L. The numbers O and D will be different and L will be at most 10000.

OUTPUT

Output M integers each on its own line – for each road, the number of different shortest paths containing it, modulo 1 000 000 007. The order of these numbers should match the order of roads in the input.

SCORING

In test cases worth 30% of points, N will be at most 15 and M will be at most 30.

In test cases worth 60% of points, N will be at most 300 and M will be at most 1000.

EXAMPLES

input

```
4 3
1 2 5
2 3 5
3 4 5
```

output

```
3
4
3
```

input

```
4 4
1 2 5
2 3 5
3 4 5
1 4 8
```

output

```
2
3
2
1
```

input

```
5 8
1 2 20
1 3 2
2 3 2
4 2 3
4 2 3
3 4 5
4 3 5
5 4 20
```

output

```
0
4
6
6
6
7
2
6
```

TASK	MJEHURIC	DATUM	ROT	SLIKAR	TREZOR	PERIODNI
input	standard input					
output	standard output					
time limit	1 second	1 second	1 second	1 second	3 seconds	5 seconds
memory limit	32 MB	32 MB	32 MB	32 MB	32 MB	32 MB
points	**40**	**40**	**70**	**100**	**120**	**130**
	500					

Goran has five wooden pieces arranged in a sequence. There is a number between 1 and 5 inscribed on every piece, so that every number appears on exactly one of the five pieces.

Goran wants to order the pieces to form the sequence 1, 2, 3, 4, 5 and does it like this:

1. If the number on the first piece is greater than the number on the second piece, swap them.
2. If the number on the second piece is greater than the number on the third piece, swap them.
3. If the number on the third piece is greater than the number on the fourth piece, swap them.
4. If the number on the fourth piece is greater than the number on the fifth piece, swap them.
5. If the pieces don't form the sequence 1, 2, 3, 4, 5, go to step 1.

Write a program that, given the initial ordering of the pieces, outputs the ordering after each swap.

INPUT

The first line contains five integers separated by single spaces, the ordering of the pieces.

The numbers will be between 1 and 5 (inclusive) and there will be no duplicates.

The initial ordering will not be 1, 2, 3, 4, 5.

OUTPUT

After any two pieces are swapped, output the ordering of the pieces, on a single line separated by spaces.

EXAMPLES

input 2 1 5 3 4 **output** 1 2 5 3 4 1 2 3 5 4 1 2 3 4 5	**input** 2 3 4 5 1 **output** 2 3 4 1 5 2 3 1 4 5 2 1 3 4 5 1 2 3 4 5

Write a program that, given a date in 2009, determines the day of week on that date.

INPUT

The first line contains two positive integers D and M separated by a space. The numbers will be a valid date in 2009.

OUTPUT

Output the day of the week on D. M. 2009. The output should be one of the words "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday" or "Sunday".

EXAMPLES

input	**input**	**input**
1 1	17 1	25 9
output	**output**	**output**
Thursday	Saturday	Friday

Damir likes to rotate. Right now he is rotating tables of letters. He wrote an R×C table onto a piece of paper. He has also chosen an angle K, a multiple of 45, and wants to rotate his table that many degrees clockwise.

It turns out this task is a bit too hard for Damir, so help him out.

INPUT

The first line contains two integers R and C separated by a space (1 ≤ R ≤ 10, 1 ≤ C ≤ 10) the number of rows and columns in Damir's table.

Each of the next R lines contains one row of Damir's table, a string of C lowercase letters.

The last line contains an integer K, a multiple of 45 between 0 and 360 (inclusive).

OUTPUT

Output Damir's table rotated K degrees clockwise, like shown in the examples. The output should contain the smallest number of rows necessary. Some rows may have leading spaces, but no rows may have trailing spaces.

EXAMPLES

input

```
3 5
damir
marko
darko
45
```

output

```
  d
 m a
d a m
 a r i
  r k r
   k o
    o
```

input

```
3 5
damir
marko
darko
90
```

output

```
dmd
aaa
rrm
kki
oor
```

input

```
5 5
abcde
bcdef
cdefg
defgh
efghi
315
```

output

```
    e
   d f
  c e g
 b d f h
a c e g i
 b d f h
  c e g
   d f
    e
```

Josip is a strange painter. He wants to paint a picture consisting of N×N pixels, where N is a power of two (1, 2, 4, 8, 16 etc.). Each pixel will be either black or white. Josip already has an idea of how each pixel will be coloured.

This would be no problem if Josip's painting process wasn't strange. He uses the following recursive process:

- If the picture is a single pixel, he colours it the way he intended.
- Otherwise, split the square into four smaller squares and then:
  1. Select one of the four squares and colour it white.
  2. Select one of the three remaining squares and colour it black.
  3. Consider the two remaining squares as new paintings and use the same three-step process on them.

Soon he noticed that it was not possible to convert all his visions to paintings with this process. Your task is to write a program that will paint a picture that differs as little as possible from the desired picture. The difference between two pictures is the number of pairs of pixels in corresponding positions that differ in colour.

INPUT

The first line contains an integer N (1 ≤ N ≤ 512), the size of the picture Josip would like to paint. N will be a power of 2.

Each of the following N lines contains N digits 0 or 1, white and black squares in the target picture.

OUTPUT

On the first line, output the smallest possible difference that can be achieved.

On the next N lines, output a picture that can be painted with Josip's process and achieves the smallest difference. The picture should be in the same format as in the input.

Note: The second part of the output (the picture) may not be unique. Any correct output will be accepted.

SCORING

In test cases worth 50% points, N will be at most 8.

EXAMPLES

input 4 0001 0001 0011 1110 **output** 1 0001 0001 0011 1111	**input** 4 1111 1111 1111 1111 **output** 6 0011 0011 0111 1101	**input** 8 01010001 10100011 01010111 10101111 01010111 10100011 01010001 10100000 **output** 16 00000001 00000011 00000111 00001111 11110111 11110011 11110001 11110000

Mirko decided to open a new business – bank vaults. A branch of the bank can be visualized in a plane, vaults being points in the plane. Mirko's branch contains exactly $L \cdot (A+1+B)$ vaults, so that each point with integer coordinates inside the rectangle with corners $(1, -A)$ and (L, B) contains one vault.

The vaults are watched by two guards – one at $(0, -A)$, the other at $(0, B)$. A guard can **see** a vault if **there are no other vaults** on the line segment connecting them.

A vault is not secure if **neither** guard can see it, secure if **only one** guard can see it and super-secure if **both** guards can see it.

Given A, B and L, output the number of insecure, secure and super-secure vaults.

INPUT

The first line contains integers A and B separated by a space ($1 \le A \le 2000$, $1 \le B \le 2000$).

The second line contains the integer L ($1 \le L \le 1\,000\,000\,000$).

OUTPUT

Output on three separate lines the numbers of insecure, secure and super-secure vaults.

SCORING

In test cases worth 50% of points, L will be at most 1000.

In test worth another 25% of points, A and B will be at most 100 (but L can be as large as one billion).

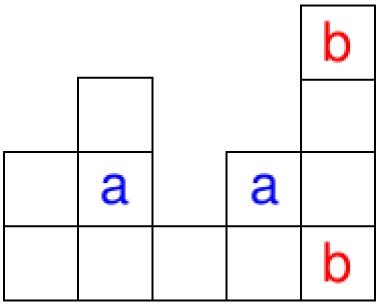
EXAMPLES

input	input	input
1 1 3	2 3 4	7 11 1000000
output	**output**	**output**
2 2 5	0 16 8	6723409 2301730 9974861

Luka is bored in chemistry class so he is staring at a large periodic table of chemical elements hanging from a wall above the blackboard. To kill time, Luka decided to make his own table completely different from the one in the classroom.

His table consists of N columns, each with some height, aligned at the bottom (see example below). After he draws the table he needs to fill it with elements. He first decided to enter the noble gases of which there are K. Luka must put them in the table so that no two noble gases are **close** to each other.

Two squares in the table are close to each other if they are in the same column or row, and all squares between them exist. In the example below, the 'a' squares are not close, but the 'b' squares are.



Write a program that, given N, K and the heights of the N columns, calculates the total number of ways for Luka to place the noble gases into the table. This number can be large, so output it modulo 1000000007.

INPUT

The first line contains the integers N and K separated by a space (1 ≤ N ≤ 500, 1 ≤ K ≤ 500), the number of columns in Luka's table and the number of noble gases.

The next line contains N positive integers, separated by spaces. These are heights of the columns from left to right. The heights will be at most 1000000.

OUTPUT

Output the number of ways for Luka to fill his table with noble gases, modulo 1000000007.

SCORING

In test cases worth 40% of points, all numbers in the input will be less than 15.

In test cases worth 70% of points, all numbers in the input will be less than 100.

EXAMPLES

input	**input**	**output**	**input**
3 3 2 1 3	4 1 1 2 3 4	5 2 2 3 1 2 4	3 2 999999 999999 999999
output	**output**	**output**	**output**
2	10	43	990979013