Problem A. Balls

Input file:	`balls.in`
Output file:	`balls.out`
Time limit:	1 second
Memory limit:	256 megabytes

There are N balls painted with not more than to M colours in a basket. Colours are numbered by integers from 1 to M, i-th ball is painted in colour C_i. Somebody performs the following sequence of steps K times:

1. take a ball out of the basket;
2. write its colour number on a sheet of paper;
3. throw this ball away.

You must count the number of different number sequences which can be written.

Input

First line of input file contains integer numbers N, M and K ($1 \le N \le 200$, $1 \le M \le N$, $1 \le K \le N$). On the next line there are N integer numbers: $C_1, \ldots, C_n$ ($1 \le C_i \le M$). All numbers in lines are separated by spaces.

Output

Output file must contain one integer number without leading zeroes — answer for the task.

Example

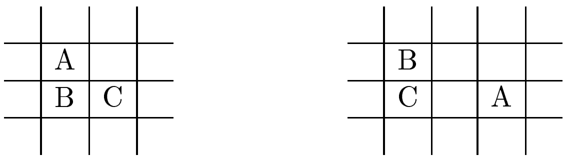
balls.in	balls.out
5 2 3 1 2 1 2 2	7

Sequences that may appear: $(1,1,2)$, $(1,2,1)$, $(2,1,1)$, $(1,2,2)$, $(2,1,2)$, $(2,2,1)$, $(2,2,2)$.

Problem B. Checkers

Input file:	`checkers.in`
Output file:	`checkers.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Peter likes to play with checkers on a chequered board. Recently he invented a new game: transform one beautiful picture composed of checkers into another beautiful picture. The following moves are allowed. Consider two following configurations of three selected cells each (let's denote the cells A, B and C):



.

If we select three fields arranged in any of the two ways shown above, you are allowed to do one of two possible moves. If cell A is occupied and cells B and C are free, it is allowed to remove checker from cell A and place two checkers into cells B and C. Vice versa, if cells B and C are occupied and cell A is free, it is allowed to remove checkers from cells B and C and place checker into cell A. Configurations can be taken in any place of the board, but can't be rotated or reflected. It is forbidden to make move if you try to remove checker from empty cell or if you try to place checker into occupied cell. Peter has such huge board that you can assume that it is infinite in any direction. Also you can assume that Peter has infinite number of checkers.

Peter decided to give riddle to his friend Vasya. He asked to transform one picture into another. Vasya tried different combinations of moves for a long time. Several times he got something simmilar to his goal, but not the same picture. Then Vasya decided that it's impossible to crack the riddle. But Peter said he has list of necessary moves.

Now Vasya asks you to write program which could help him solve Peter's puzzle.

Input

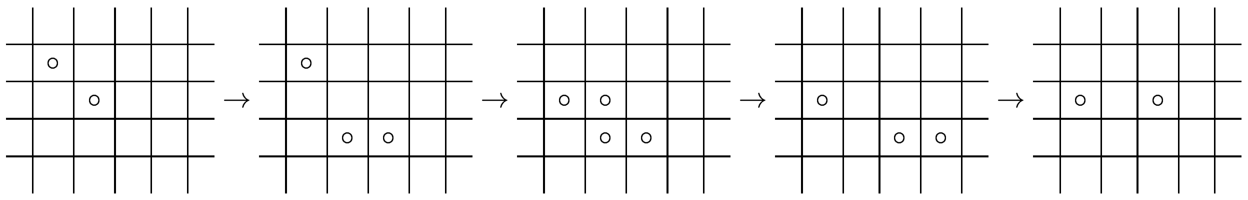
The first line of the input file contains N and M ($1 \leq N, M \leq 10$) — dimensions of the first picture. Following N lines describe the picture. Each line contain M symbols '`.`' or '`#`'. '`.`' means that cell is empty and '`#`' means that cell is occupied by checker. Description of the second picture in the same format follows immediately after first picture.

Output

Output sequence of moves that transforms first picture into second. Every line of ouput should contain three integer numbers — coordinates of cell A and number of configuration (1 or 2) for next move. Left bottom cell of each picture has coordinates $(0, 0)$ and right top cell has coordinates $(M - 1, N - 1)$. This cells are located in the same cell on the plane.

Example

checkers.in	checkers.out
2 2	1 0 1
#.	0 1 1
.#	3 -1 2
1 3	2 0 1
#.#	



One can prove that for any correct testcase there exists a sequence of less than 70 000 moves which solves this testcase.

Problem C. Checkers 2

Input file:	`checkers2.in`
Output file:	`checkers2.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Read the **Checkers** problem.

With your help Vasya solved Peter's problem without any troubles. Vasya asked himself why he can't solve the problem without your help. Maybe one of his moves was incorrect. Vasya doesn't remember his moves. He remembers only initial and final positions. He wonders whether it is possible to get one position from the other.

Input

The first line of the input file contains N and M ($1 \le N, M \le 400$) — dimentions of the first picture. Following N lines describe the first picture. Each line contain M symbols '`.`' or '`#`'. '`.`' means that cell is empty and '`#`' means that cell is occupied by checker. Description of the second picture in the same format follows immediately after first picture.

Output

Print "`Yes`" if it is possible to get one picture from another and "`No`" otherwise.

Examples

checkers2.in	checkers2.out

`1 2`
`.#`
`1 1`
`#`
	`No`

`2 2`
`#.`
`.#`
`1 3`
`#.#`
	`Yes`

Problem D. Contest

Input file:	`contest.in`
Output file:	`contest.out`
Time limit:	1 second
Memory limit:	256 megabytes

According to rules of some programming competition, N contestants are randomly distributed into M groups so that there is equal number of people in each group. All groups solve same tasks. After end of the contest they can see results. The results table is a table which consists of N rows containing name and points of contestant. Table is sorted in descending order by points. The probability for two contestants to get same number of points is very low, so we think that it's impossible.

Also, rows of group winners are highlighted. Group winner is a contestant which got most points in his group. It's obvious that other contestants from this group will be located lower in results table. Number of selected rows is equal to groups number.

Find the number of ways one can distribute contestants to groups if he knows indices of highlighted rows of the table.

Input

First line of input file contains integer numbers N, M, P ($1 \le N, M \le 100$, M divides N, $2 \le P \le 10^9$). Next M lines contain one number each — highlited rows indices. Rows are numbered from 1 to N starting from the top.

Output

If there are no ways to distribute contestants, output -1. In other case, output one number — answer for the task modulo P.

Examples

`contest.in`	`contest.out`

`4 2 1000`
`1`
`2`
	`2`

`4 2 1000`
`1`
`4`
	`-1`

Problem E. K-th Divisor

Input file:	`divisor.in`
Output file:	`divisor.out`
Time limit:	5 seconds
Memory limit:	256 megabytes

Find K-th smallest divisor of number N.

Input

The first line contains number of test cases M $(1 \le M \le 30\,000)$. Each of the next M lines contains one test case: two integers N and K $(1 \le N \le 1\ 000\ 000\ 000, 1 \le K \le$ amount of divisors of number $N)$.

Output

For each test case print one number — K-th divisor of number N on a single line.

Example

divisor.in	divisor.out

4
10 1
10 2
10 3
10 4
	1
	2
	5
	10

Problem F. Fence

Input file:	`fence.in`
Output file:	`fence.out`
Time limit:	7 seconds
Memory limit:	256 megabytes

Peter has N poles in his vegetable garden. He wants to keep only four poles, all other he will dig out. But he has one restriction — these four poles must form vertices of parallelogram with positive area. Peter wants to know how many ways there are to do this.

Input

There is one integer N ($1 \le N \le 2000$) in the first line of input file. Each of the next N lines contains two integers x_i, y_i ($1 \le x_i, y_i \le 1\,000\,000\,000$) — coordinates of i-th pole.

Output

Print the ways count.

Example

fence.in	fence.out

6
1 1
2 1
3 1
1 2
2 2
3 2
	5

Problem G. Good Graphs

Input file:	`good.in`
Output file:	`good.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Alex defined *good graphs*:

- Single vertex is a *good graph*.
- If two *good graphs* have no common vertex then their union is a *good graph*.
- If G is a *good graph* then $\overline{G}$ (complement of G) is a *good graph*.

Try to solve the problem of finding maximal weighted clique in a *good graph*.

Input

The first line of contains the integer N ($1 \le N \le 500$) — number of vertices in the *good graph* G.

The next N lines contain adjacency matrix of G.

Each of last N lines contains the integer w_i ($1 \le w_i \le 1000$) — the weight of ith vertex.

Output

In the single line of the output file print the maximal weight of clique of graph G.

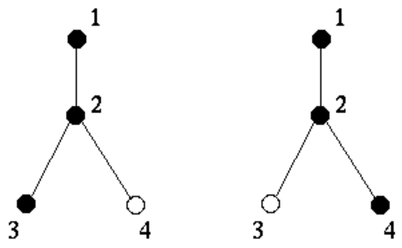
Example

good.in	good.out
4 0000 0011 0101 0110 100 1 2 3	100

Problem H. Graph

Input file:	`graph.in`
Output file:	`graph.out`
Time limit:	1 second
Memory limit:	256 megabytes

Let's consider undirected graph with N vertices and M edges. How many different ways are there to paint it, if there are only K colours? You need not use all colours in one painting. Two paintings are considered the same if there is such a renumbering of vertices of one painting that leaves the list of edges unchanged, and the colour of its i-th vertex is the same as the colour of i-th vertex of other painting for each i. For example, paintings on picture are the same (renumbering: $1 \to 1, 2 \to 2, 3 \to 4, 4 \to 3$).



Input

First line of input file contains three integer numbers: N, M and K ($1 \le N \le 9$, $1 \le M \le 100$, $1 \le K \le 10$). Next M lines contain two integers each — graph edges. Graph may contain parallel edges and loops. Numbers in lines are separated by spaces.

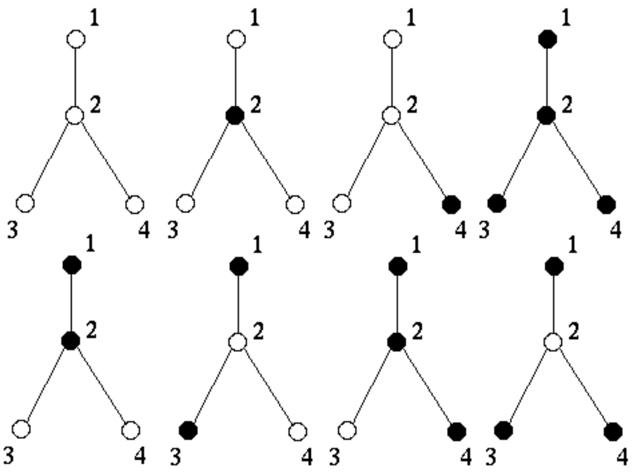
Output

Output file must contain one integer number K — answer for the task.

Example

graph.in	graph.out

4 3 2
1 2
2 3
2 4
	8



Problem I. Matrix

Input file:	`matrix.in`
Output file:	`matrix.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Find number of nonsingular (determinant is non-zero) 2×2 matrices with elements from $\mathbb{Z}_n$.

Input

The first line of the input file contains n and m ($1 \le n \le 10^{14}$, $1 \le m \le 10^9$).

Output

In the single line of the output file print one number — the answer modulo m.

Example

`matrix.in`	`matrix.out`
`3 1000000000`	`48`

Problem J. Permutations

Input file:	`permutations.in`
Output file:	`permutations.out`
Time limit:	1 second
Memory limit:	256 megabytes

Let's take two permutations: $A = (a_1, a_2, ...a_N)$ and $B = (b_1, b_2, ...b_N)$. We can define their product as permutation $C = A \times B = (c_1, c_2, ...c_N)$, where $c_i = a_{b_i}$. For example, for $A = (3, 5, 1, 2, 4)$ and $B = (2, 3, 5, 4, 1)$, their product is $C = (5, 1, 4, 2, 3)$.

It is well-known that in common case multiplication of permutations is not commutative (i.e. $A \times B \neq B \times A$). Example:

$(3, 5, 1, 2, 4) \times (2, 3, 5, 4, 1) = (5, 1, 4, 2, 3)$

$(2, 3, 5, 4, 1) \times (3, 5, 1, 2, 4) = (5, 1, 2, 3, 4)$

But there are some pairs, multiplication of which is commutative. Example:

$(3, 4, 2, 5, 1) \times (4, 1, 5, 3, 2) = (5, 3, 1, 2, 4)$

$(4, 1, 5, 3, 2) \times (3, 4, 2, 5, 1) = (5, 3, 1, 2, 4)$

You are given permutation A. Count number of such permutations B that $A \times B = B \times A$, and output it modulo P.

Input

First line of input file contains 2 integer numbers: N and P ($1 \leq N \leq 10^5$, $2 \leq P \leq 10^9$). On the second line there are N different integer numbers: $a_1, a_2, ...a_N$ — permutation A ($1 \leq a_i \leq N$). All numbers in lines are separated by spaces.

Output

Output file must contain one integer number K — answer for the task ($0 \leq K \leq P - 1$).

Example

`permutations.in`	`permutations.out`

`5 100`
`3 4 2 5 1`
	`5`

Permutations for example:

1 2 3 4 5

2 5 4 1 3

3 4 2 5 1

4 1 5 3 2

5 3 1 2 4

Problem K. Segments

Input file:	`segments.in`
Output file:	`segments.out`
Time limit:	3 seconds
Memory limit:	256 megabytes

There are N segments on the plane. Each segment is either vertical or horizontal. Find the amount of segment pairs that have at least one common point.

Input

First line contains one number N ($1 \le N \le 100\,000$). Each of the next N lines contains four integers — coordinates of the segment's endpoints x_1 y_1 x_2 y_2 ($-1\,000\,000 \le x_1, y_1, x_2, y_2 \le 1\,000\,000$).

Output

In the single line of the output file print the amount of segment pairs that have at least one common point.

Example

`segments.in`	`segments.out`

4
0 0 0 0
0 0 1 0
0 -1 0 1
1 -1 1 1
	4

Problem L. Subsequence

Input file:	`subsequence.in`
Output file:	`subsequence.out`
Time limit:	1 second
Memory limit:	256 megabytes

Consider lexicographically ordered set $\mathcal{S}$ of different ascending subsequences of a given sequence of N integer numbers. Your task is to find K-th element of this set in lexicographical order.

Input

First line of input file contains two integer numbers: N and K ($1 \le N \le 60$, $1 \le K \le |\mathcal{S}|$). On the second line there are N integer numbers in interval from 1 to 10^9, inclusive. Numbers in lines are separated by spaces. It is guaranteed that such subsequence exists.

Output

First line of output file must contain one integer number M — the number of elements in the resulting subsequence. Next line must contain M integers separated by spaces — the subsequence itself.

Example

subsequence.in	subsequence.out
3 2 1 1 2	2 1 2

Problem M. Sum of Cyclic Shifts

Input file:	`sum.in`
Output file:	`sum.out`
Time limit:	3 seconds
Memory limit:	256 megabytes

Let A be an integer and $a_1a_2\ldots a_n$ be its decimal notation. Let us define $\text{Shift}(a_1a_2\ldots a_n) = a_2a_3\ldots a_na_1$. Let $A_1 = A$ and $A_i = \text{Shift}(A_{i-1})$ for $i = \overline{2, n}$. Your task is to find a minimal divisor $d \neq 1$ of $\sum\limits_{i=1}^{n} A_i$.

Input

A single line contains $1 < A < 10^{1\,000\,000}$. A does not contain zero digit.

Output

Print the required number d.

Example

sum.in	sum.out
12345	3

Problem N. Byteland Tour

Input file:	`tour.in`
Output file:	`tour.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Mister X is going to visit Byteland and wants to make a tour of the country. There are some bidirectional roads between the cities. All roads connect different pairs of the cities. There is no road that connects a city with itself.

Mister X hasn't already decided which city would be the first in his tour, though he has decided how he would move from one city to another. When he is in the city A he chooses any nonvisited city that can be directly reached from A, and moves to it. If there is no such city, he finishes his tour. Mister X wants to know if any of his possible routes (independent from choosing starting city and next nonvisited cities) contains all the cities. Your task is to help him.

Input

The first line of the input file contains two integers N and M ($1 \le N \le 100\,000$, $0 \le M \le 200\,000$): the number of cities and the number of roads in Byteland. Each of the next M lines contains two integers: the numbers a_i, b_i ($1 \le a_i, b_i \le N$) of two cities connected by road. All the roads connect different pairs of the cities.

Output

In the single line of the output file print "`YES`" if every Mister X's route contains all N cities, otherwise print "`NO`".

Examples

tour.in	tour.out

3 3
1 2
2 3
3 1
	YES

3 2
1 2
2 3
	NO