Problem A. Amazing Trick

Time limit:	3 seconds
Memory limit:	1024 megabytes

Alice is a magician and she creates a new trick. She has n cards with different numbers from 1 to n written on them. First, she asks an audience member to shuffle the deck and put cards in a row. Let's say the i-th card from the left has the number a_i on it.

Then Alice picks two permutations p and q. There is a restriction on p and q — **permutations can't have fixed points**. Which means $\forall i : p_i \neq i \textit{ and } q_i \neq i$.

After permutations are chosen, Alice shuffles the cards according to them. Now the i-th card from the left is the card $a[p[q[i]]$. The trick is considered successful if i-th card from the left has the number i on it after the shuffles.

Help Alice pick the permutations p and q or say it is not possible for the specific starting permutation a.

Input

The first line of the input contains the number of tests t ($1 \leq t \leq 10^5$).

Each test is described in two lines. The first line contains one integer n — the number of cards ($1 \leq n \leq 10^5$). The second line contains n integers a_i — the initial permutation of the cards ($1 \leq a_i \leq n$; $\forall i \neq j : a_i \neq a_j$).

It is guaranteed that the sum of n over all tests does not exceed 10^5.

Output

Print the answer for each test case in the same order the cases appear in the input.

For each test case, print "`Impossible`" in a single line, if no solution exists.

Otherwise, print "`Possible`" in the first line, and in the following two lines print permutations p and q.

Example

standard input	standard output

```
4
2
2 1
3
1 2 3
4
2 1 4 3
5
5 1 4 2 3
```
	```
	Impossible
	Possible
	3 1 2
	2 3 1
	Possible
	3 4 2 1
	3 4 2 1
	Possible
	4 1 2 5 3
	3 1 4 5 2
	```

Problem B. BinCoin

Time limit:	3 seconds
Memory limit:	1024 megabytes

There are n employees in the BinCoin company numbered from 1 to n. The subordination structure in this company is a rooted tree. In other words:

- There is one CEO in the company — the main boss.
- Each other employee has exactly one direct superior.
- There are no cycles in the subordination structure.

Moreover, due to the inexplicable love of the CEO of BinCoin for all the binary stuff, the subordination structure in the company is a **binary** rooted tree. That means each employee is directly superior to exactly zero or two other employees.

In the CEO's opinion, working in this company is almost as dangerous as in mines. So, employees should sign the waiver of claims sometimes. This process happens in the following way. Initially, CEO takes the journal, then recursively the following procedure is performed:

- If an employee that holds the journal does not have any subordinates, they sign the waiver in the journal and give it back to their superior. The procedure stops if that was the CEO, who has no superior.
- Otherwise
  - they choose one of two of their direct subordinates uniformly at random and give the journal to one of them;
  - when they get the journal back, they sign it;
  - and then they give it to another direct subordinate;
  - when they get it back again, they give it back to their superior. The procedure stops if that was the CEO, who has no superior.

All random choices are independent.

One day, the CEO realized that they could not remember the subordination tree. Fortunately, they have the journal with k records. Each record is a sequence of employees in the order they've signed in a journal.

Help CEO restore the subordination tree.

Input

The first line contains two integers n and k — the number of employees and the number of records in the journal ($1 \le n \le 999$; $50 \le k \le 100$).

Each of the next k lines contains a permutation of integers from 1 to n — the order of employees in the corresponding record.

It is guaranteed that the input was obtained as described in the statement with a real random choice.

Output

Output n integers p_i. If i is a CEO, then p_i should be -1. Otherwise, p_i should be the index of the direct superior of i-th employee.

Your output should correspond to a binary rooted tree. If there are several trees satisfying the input, you can output any one of them.

Examples

standard input	standard output
3 50	2 -1 2
1 2 3 1 2 3 3 2 1 1 2 3	
3 2 1 1 2 3 1 2 3 1 2 3	
1 2 3 3 2 1 1 2 3 3 2 1	
1 2 3 3 2 1 1 2 3 3 2 1	
1 2 3 1 2 3 3 2 1 1 2 3	
3 2 1 1 2 3 3 2 1 1 2 3	
1 2 3 3 2 1 1 2 3 1 2 3	
1 2 3 1 2 3 3 2 1 1 2 3	
3 2 1 3 2 1 1 2 3 3 2 1	
1 2 3 3 2 1 3 2 1 1 2 3	
1 2 3 3 2 1 1 2 3 3 2 1	
3 2 1 3 2 1 1 2 3 1 2 3	
3 2 1 3 2 1	
5 60	5 4 4 5 -1
2 4 3 5 1 1 5 2 4 3 1 5 2 4 3	
1 5 2 4 3 1 5 3 4 2 1 5 3 4 2	
1 5 3 4 2 1 5 3 4 2 1 5 3 4 2	
3 4 2 5 1 2 4 3 5 1 1 5 2 4 3	
3 4 2 5 1 2 4 3 5 1 2 4 3 5 1	
1 5 2 4 3 3 4 2 5 1 3 4 2 5 1	
1 5 2 4 3 2 4 3 5 1 1 5 2 4 3	
1 5 3 4 2 3 4 2 5 1 1 5 3 4 2	
1 5 2 4 3 1 5 3 4 2 1 5 2 4 3	
2 4 3 5 1 2 4 3 5 1 2 4 3 5 1	
2 4 3 5 1 2 4 3 5 1 1 5 2 4 3	
1 5 3 4 2 1 5 2 4 3 3 4 2 5 1	
1 5 3 4 2 3 4 2 5 1 3 4 2 5 1	
1 5 2 4 3 2 4 3 5 1 1 5 2 4 3	
1 5 3 4 2 2 4 3 5 1 2 4 3 5 1	
1 5 2 4 3 1 5 2 4 3 1 5 2 4 3	
1 5 2 4 3 1 5 2 4 3 3 4 2 5 1	
3 4 2 5 1 3 4 2 5 1 1 5 2 4 3	
1 5 3 4 2 1 5 3 4 2 2 4 3 5 1	
3 4 2 5 1 1 5 2 4 3 3 4 2 5 1	

Note

In order to fit on the page, several consecutive lines in the examples were joined into one. The real inputs follow the input description.

Problem C. Cactus Meets Torus

Time limit:	3 seconds
Memory limit:	1024 megabytes

Alice has a nice cactus graph that she wanted to place on a sheet of paper. Eve threatens to take one cycle of this cactus and cut the paper along all edges on this cycle. This way, the sheet of paper will be divided in two parts, and Alice will be upset. Luckily, Barbara just gave Alice a paper torus — a sheet of paper where top and bottom edges are connected as well as left and right edges are connected without twisting. On torus, you can sometimes cut paper along all edges on a cycle, but it would still remain in one piece. Help Alice to determine if she can place her cactus on a torus such that Eve cannot cut paper along one cycle dividing the torus into two unconnected pieces.

Cactus is a connected undirected graph in which every edge lies on at most one simple cycle. Intuitively, cactus is a generalization of a tree where some cycles are allowed. Multiedges (multiple edges between a pair of vertices) and loops (edges that connect a vertex to itself) are not allowed in a cactus.

We say that a graph is placed on a sheet of paper if each vertex is a point on this sheet, each edge is a segment between points corresponding to its vertices, and these segments only intersect at their ends. On torus segments can go through sheet edges any number of times.

Input

The input consists of one or more independent test cases.

The first line of each test case contains two integers n and m ($1 \le n \le 10^5$; $0 \le m \le 10^5$), where n is the number of vertices in the graph. Vertices are numbered from 1 to n. The edges of the graph are represented by a set of edge-distinct paths, where m is the number of such paths.

Each of the following m lines contains a path in the graph. A path starts with an integer s_i ($2 \le s_i \le 1000$) followed by s_i integers from 1 to n. These s_i integers represent vertices of a path. Adjacent vertices in a path are distinct. The path can go through the same vertex multiple times, but every edge is traversed exactly once in the whole test case. There are no multiedges in the graph (there is at most one edge between any two vertices).

The last line of the input after all test cases contains two zeros. It does **not** define a test case. It just marks the end of the input and does not require any output.

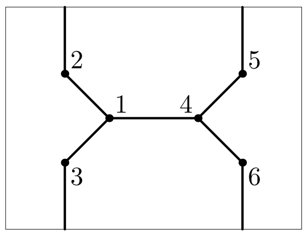
All graphs in the input are cacti. The total sum of all values of n and the total sum of all values of m throughout the input both do not exceed 10^5.

Output

Print the answer for each test case in the same order the cases appear in the input. For each test case, print a single line with "Yes" if you can place this cactus on a torus or print "No" otherwise.

Example

standard input	standard output	picture

`6 1`
`8 1 2 3 1 4 5 6 4`
`10 2`
`9 1 2 3 1 10 4 5 6 4`
`5 7 8 9 7 10`
`0 0`
	`Yes`
	`No`
		

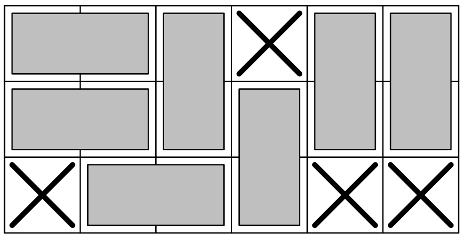
Note

One way to place the cactus from the first case on a torus is shown on the picture.

Problem D. Dominoes

Time limit:	3 seconds
Memory limit:	1024 megabytes

Dora likes to play with dominoes. She takes $n \times m$ table, marks some cells as occupied, and then tries to fill all unoccupied cells with 2×1 dominoes.



Her little brother Dani loves to play pranks on his older sister. So when she is away, he marks two more unoccupied cells as occupied. He wants to do it in such a way that it will be impossible to fill all unoccupied cells with dominoes.

Help Dani to count the number of ways he can select these two cells. Since Dani can only count to one million, if this number of ways is x, output $\min(x, 10^6)$.

Input

The first line contains integers n and m ($1 \le n, m \le 1000$). Next n lines contain m characters each — the initial state of the table. Character "#" corresponds to an occupied cell, and character "." corresponds to an unoccupied cell. It is guaranteed that there are at least two unoccupied cells, and that it is possible to fill all unoccupied cells with dominoes.

Output

Let x be the number of ways Dani can mark two cells in such a way that it will be impossible to fill all unoccupied cells with dominoes.

Print one integer $\min(x, 10^6)$.

Examples

standard input	standard output
3 6 ...#.. ...... #...##	52

2 2
..
..
	2

2 2 #. #.	0

Problem E. Easy Assembly

Time limit:	3 seconds
Memory limit:	1024 megabytes

Emma loves playing with blocks. She has several cubic blocks of the same size that are numbered with **distinct** integers written on them. She assembles towers from those blocks by stacking them vertically.

A configuration of her game is a set of towers that she has assembled from the blocks. Emma can perform two kinds of operations on a configuration of towers:

- **Split** any tower with more than one block in it by taking any number of blocks from the top of the tower and moving them to a new tower keeping their order, so that the top block of the old tower becomes the top block of the new tower. As a result of this operation, the number of towers increases by one.
- **Combine** any two towers by moving blocks from one tower on top of the other tower in the same order. As a result of this operation, the number of towers decreases by one.

Emma wants to stack all the blocks into a single tower so that all blocks come in order sorted by the numbers — from the block with the minimal number at the top to the block with the maximal number at the bottom. Emma wants to do as little of splitting and combining operations as possible. Your task is to find the minimal number of operations she has to make and output how many splits and combines are needed.

Input

The first line of the input file contains an integer n ($1 \le n \le 10\,000$) — the number of towers in the initial configuration. Next n lines describe towers. Each tower i is described by a line that starts with the number k_i ($k_i \ge 1$; $\sum_1^n k_i \le 10\,000$) — the number of blocks in the tower, followed by k_i numbers $b_{i,j}$ ($1 \le b_{i,j} \le 10^9$) — numbers written on the blocks in the i-th tower, listed from top to bottom. All block numbers listed in the input are different.

Output

Output a line with two integers s and c — the number of split and combine operations Emma should make to get a single tower with blocks sorted by their numbers, so that the total number of operations is minimized.

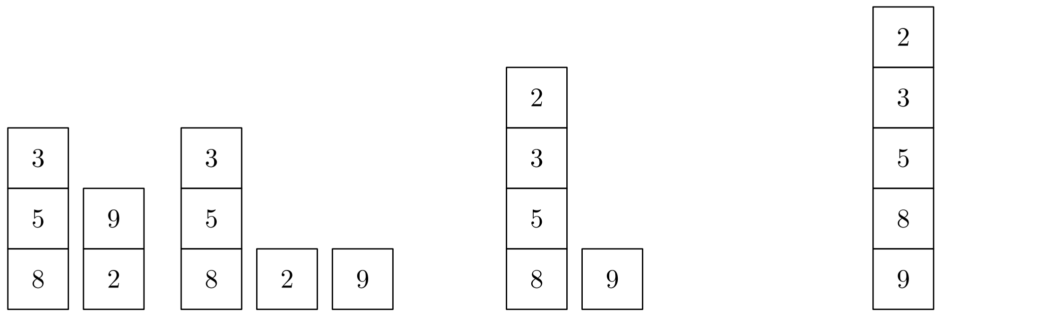
Example

standard input	standard output

2
3 3 5 8
2 9 2
	1 2

Note

The example needs the following operations (1 split and 2 combines).



Initial Split last Combined 2nd onto 1st Combined 1st onto 2nd

Problem F. Football

Time limit:	3 seconds
Memory limit:	1024 megabytes

Scientists are researching an impact of football match results on the mood of football fans. They have a hypothesis that there is a correlation between the number of draws and fans' desire to watch football matches in the future.

In football, two teams play a match. The teams score goals throughout a match. A score "$x : y$" means that the team we observe scored x goals and conceded y goals. If $x = y$, then the match ends in a draw. If $x > y$, then the observed team wins, and if $x < y$, then it loses.

To find out if there is a correlation, the scientists gathered information about the results of teams in lower leagues. The information they found is the number of matches played by the team (n), the number of goals scored in these matches (a), and the number of goals conceded in these matches (b).

You are given this information for a single team. You are asked to calculate the minimum number of draws that could have happened during the team's matches and provide a list of match scores with the minimum number of draws.

Input

The first line contains an integer n — the number of matches played by the team ($1 \le n \le 100$). The second line contains an integer a — the total number of goals scored by the team in all n matches ($0 \le a \le 1000$). The third line contains an integer b — the total number of goals conceded by the team in all n matches ($0 \le b \le 1000$).

Output

In the first line, print a single integer d — the minimum number of draws.

In the following n lines, print a list of match scores, each line in the format "$x : y$", where x is the number of goals scored in the match, and y – the number of goals conceded, so that exactly d of these matches have ended in a draw. In case multiple such lists of match scores exist, print any of them.

Examples

standard input	standard output
3 2 4	0 1:0 1:2 0:2
1 2 2	1 2:2
4 0 7	0 0:1 0:2 0:1 0:3
6 3 1	2 0:0 1:0 0:0 0:1 1:0 1:0

Problem G. Game of Questions

Time limit: 5 seconds
Memory limit: 1024 megabytes

Genie is taking part in an intellectual game. The game consists of n questions, and there are m participants numbered from 1 to m. Genie is the participant number 1.

For each question i and participant j, it is known whether the participant will answer the question correctly or not.

The goal of the game is to be the last participant staying in the game.

The game is conducted as follows. First, all n questions get shuffled uniformly at random (all $n!$ permutations are equally likely). Then, the questions are asked one by one. Each participant answers the question. If all participants still in the game answer the question correctly, or if all of them answer the question incorrectly, nothing happens. Otherwise, those participants who answer the question incorrectly lose and leave the game.

After all n questions are asked, all participants who are still in the game are declared to be the winners.

What is the probability that Genie will win the game?

Input

The first line contains two integers n and m — the number of questions and the number of participants ($1 \le n \le 2 \cdot 10^5$; $2 \le m \le 17$).

The i-th of the next n lines contains m characters $s_{i,1}, s_{i,2}, \ldots, s_{i,m}$. Character $s_{i,j}$ is '1' if participant j answers question i correctly or '0' otherwise.

Output

Print the probability that Genie will win the game. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-9}.

Examples

standard input	standard output
1 5 11010	1.0000000000000000

3 3
011
101
110
	0.3333333333333333

6 4
1011
0110
1111
0110
0000
1101
	0.1666666666666667

Note

In the first example, there is a single question and Genie will answer it correctly, thus winning the game (along with participants 2 and 4).

In the second example, one participant will leave after the first asked question, and another participant will leave after the second asked question. Each participant will win with probability $\frac{1}{3}$.

Problem H. Hot and Cold

Time limit:	3 seconds
Memory limit:	1024 megabytes

This is an interactive problem.

After emigrating to another country, little Hanna came across the fact that playing "Hot and cold" became more difficult: she does not understand the hint phrases! You are to write a helping program for her.

The playground where the game happens is a rectangle, and the treasure is hidden in some point with integer coordinates between 0 and 10^6, inclusive. Hanna visits several points with valid coordinates. If the visited point contains the hidden treasure, she receives the phrase "Found!" in the local language. Fortunately, the intonations allow her to recognize this phrase — in this problem it will be denoted by an exclamation mark.

Otherwise, for each visited point except for the first one, Hanna receives a phrase in the local language stating whether she is now "Closer", "Further", or "At the same distance" from the treasure, compared to the previous point. The distances are measured in Euclidean metric. After the first visited point, in case of no treasure there, Hanna receives the phrase "Not found" in the local language.

Your program must help Hanna find the treasure by visiting **at most 64 points**.

Interaction Protocol

Your program should print each visited point in a line with two coordinates between 0 and 10^6, inclusive, and flush the output after each one.

For each visited point, the testing system writes one of the phrases: "Found!", "Closer", "Further", "At the same distance", or "Not found" in some language, consistent throughout the game. The first phrase ends with an exclamation mark, all other phrases do not contain exclamation marks. All phrases contain only Latin letters, spaces and exclamation marks, have lengths between 2 and 30, inclusive, don't start or end with a space, and are pairwise distinct. The phrases are not necessarily coming from any real human language.

After receiving a phrase with an exclamation mark, your program must halt without printing anything else.

Examples

standard input	standard output
	`500 200`
`Tabilmadi`	
	`560 230`
`Daha yakin`	
	`566 240`
`Daha yakin`	
	`566 238`
`Sama distanco`	
	`30 239`
`Dalej`	
	`566 239`
`Znaydeno!`	

standard input	standard output
	`777777 777777`
`You are lucky today!`	

Problem I. Interactive Factorial Guessing

Time limit:	3 seconds
Memory limit:	1024 megabytes

Oh no, this wicked jury hides something from you again, and you need to guess it interactively.

This time, you need to find an integer n. To do that, you can make at most 10 queries of the form "What is the k-th decimal digit of the product of all integers from 1 to n (also known as factorial and denoted as $n!$)?".

Interaction Protocol

In the first line, there is an integer t ($1 \le t \le 100$) — the number of tests you shall process.

For each test, the integer n is chosen in advance. The length of $n!$ is at most 20 000, so $1 \le n \le 5982$.

You can make **at most 10 queries** of the form "`?` k" ($0 \le k < 20\,000$). In response to the query, you will get a single digit — the k-th decimal digit of $n!$ (the response is between 0 and 9 inclusive). Digits are numbered from 0, starting with the least significant digit. If $n!$ is too short, and there is no k-th digit, then 0 is returned.

After your program finds the value of n it shall answer with "`!` n". If the answer is correct, then you will receive "`YES`" and should proceed to the next test or terminate if it was the last one. If the answer is not correct, or you are trying to guess, and there are several possible answers consistent with the information you have received, you will get "`NO`". In that case, your submission will receive "`Wrong answer`" verdict and your code shall terminate immediately.

Example

standard input	standard output
2	
	? 0
1	
	! 1
YES	
	? 0
0	
	? 19997
2	
	! 5982
YES	

Problem J. Jumbled Trees

Time limit:	3 seconds
Memory limit:	1024 megabytes

You are given an undirected connected graph with n vertices and m edges. Each edge has an associated counter, initially equal to 0. In one operation, you can choose an arbitrary spanning tree and add any value v to all edges of this spanning tree.

Determine if it's possible to make every counter equal to its target value x_i modulo prime p, and provide a sequence of operations that achieves it.

Input

The first line contains three integers n, m, and p — the number of vertices, the number of edges, and the prime modulus ($1 \le n \le 500$; $1 \le m \le 1000$; $2 \le p \le 10^9$, p is prime).

Next m lines contain three integers u_i, v_i, x_i each — the two endpoints of the i-th edge and the target value of that edge's counter ($1 \le u_i, v_i \le n$; $0 \le x_i < p$; $u_i \neq v_i$).

The graph is connected. There are no loops, but there may be multiple edges between the same two vertices.

Output

If the target values on counters cannot be achieved, print `-1`.

Otherwise, print t — the number of operations, followed by t lines, describing the sequence of operations. Each line starts with integer v ($0 \le v < p$) — the counter increment for this operation. Then, in the same line, followed by $n-1$ integers e_1, e_2, $\dots e_{n-1}$ ($1 \le e_i \le m$) — the edges of the spanning tree.

The number of operations t should not exceed $2m$. You don't need to minimize t. Any correct answer within the $2m$ bound is accepted. You are allowed to repeat spanning trees.

Examples

standard input	standard output

3 3 101
1 2 30
2 3 40
3 1 50
	3
	10 1 2
	20 1 3
	30 2 3

2 2 37
1 2 8
1 2 15
	2
	8 1
	15 2

5 4 5
1 3 1
2 3 2
2 5 3
4 1 4
	-1

Problem K. King's Puzzle

Time limit:	3 seconds
Memory limit:	1024 megabytes

King Kendrick is a sovereign ruler of Kotlin Kingdom. He is getting ready for the next session of the government. Kotlin Kingdom consists of n cities. These cities need to be connected by several bidirectional roads. Since ministries are responsible for aspects of safety and comfort of the kingdom's residents, some of them have made the following requirements:

- "All the cities should be connected by new roads" — Ministry of Transport and Digital Infrastructure.
- "There may not be a loop road — a road that connects a city with itself" — Ministry of Environment.
- "There should be at most one road between a pair of cities" — Treasury Department.
- "If a_i is the number of roads connected to i-th city, then the set $\{a_1, \ldots, a_n\}$ should consist of exactly k distinct numbers" — Ministry of ICPC.

King Kendrick has issues with the requirements from the Ministry of ICPC. He asks you to help him. Find any set of roads that suits all the requirements above or say that it is impossible.

Input

The only line of the input consists of two integers n and k ($1 \le k \le n \le 500$).

Output

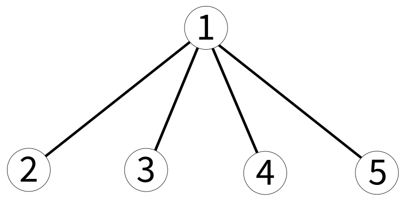
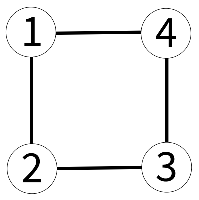
If it is impossible to satisfy all the requirements, output "NO" in the only line.

Otherwise, output "YES" in the first line.

Output m — the number of roads ($0 \le m \le \frac{n \cdot (n-1)}{2}$) in the second line.

Next m lines should contain pairs of integers a and b — the cities to connect by a road ($1 \le a, b \le n$).

Examples

standard input	standard output	notes
5 2	YES 4 1 2 1 3 1 4 1 5	 City 1 has four roads connected to it while other cities have exactly one.
4 1	YES 4 1 2 2 3 3 4 4 1	 Every city has exactly two roads connected to it.

Problem L. Lisa's Sequences

Time limit:	5 seconds
Memory limit:	1024 megabytes

Lisa loves playing with the sequences of integers. When she gets a new integer sequence a_i of length n, she starts looking for all *monotone* subsequences. A monotone subsequence $[l, r]$ is defined by two indices l and r ($1 \le l < r \le n$) such that $\forall i = l, l+1, \ldots, r-1 : a_i \le a_{i+1}$ or $\forall i = l, l+1, \ldots, r-1 : a_i \ge a_{i+1}$.

Lisa considers a sequence a_i to be *boring* if there is a monotone subsequence $[l, r]$ that is as long as her boredom threshold k, that is when $r - l + 1 = k$.

Lucas has a sequence b_i that he wants to present to Lisa, but the sequence might be boring for Lisa. So, he wants to change some elements of his sequence b_i, so that Lisa does not get bored playing with it. However, Lucas is lazy and wants to change as few elements of the sequence b_i as possible. Your task is to help Lucas find the required changes.

Input

The first line of the input contains two integers n and k ($3 \le k \le n \le 10^6$) — the length of the sequence and Lisa's boredom threshold. The second line contains n integers b_i ($1 \le b_i \le 99\,999$) — the original sequence that Lucas has.

Output

On the first line output an integer m — the minimal number of elements in b_i that needs to be changed to make the sequence not boring for Lisa. On the second line output n integers a_i ($0 \le a_i \le 100\,000$), so that the sequence of integers a_i is not boring for Lisa and is different from the original sequence b_i in exactly m positions.

Examples

standard input	standard output
5 3 1 2 3 4 5	2 1 0 3 0 5
6 3 1 1 1 1 1 1	3 1 100000 0 1 0 1
6 4 1 1 4 4 1 1	1 1 1 4 0 1 1
6 4 4 4 4 2 2 2	2 4 4 0 2 0 2
6 4 4 4 4 3 4 4	1 4 4 100000 3 4 4
8 4 2 1 1 3 3 1 1 2	2 2 1 1 3 0 1 0 2
10 4 1 1 1 2 2 1 1 2 2 1	2 1 1 100000 2 2 100000 1 2 2 1
7 5 5 4 4 3 4 4 4	0 5 4 4 3 4 4 4
10 10 1 1 1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1 0 1