Problem A. Shashlik Cooking

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

More then anything, Miroslav loves to watch burning fire, eat tasty dinner and participate in programming competitions. So it's no surprise that this task is about shashlik, Miroslav's favourite food. To cook shashlik, Miroslav uses a metal box without the top face, called mangal. In this box he burns wood, and when the wood becomes charcoal, he starts cooking shashlik. He puts metal sticks, called skewers, with beaded pieces of meat, fish, mushrooms, or vegetables, depending on his mood, parallel to each other onto the top open face of manglal. The art of cooking shashlik includes the right timing of turning over the skewers in order to heat the food evenly.

This time Miroslav laid out n skewers parallel to each other, and enumerated them with consecutive integers from 1 to n in order from left to right. For better cooking, he puts them quite close to each other, so when he turns skewer number i, it leads to turning skewers number $i - 1$ and $i + 1$ (if they exist). They do not make their neighbors turn. So if $n = 6$ and Miroslav turns skewer number 3, then skewers with numbers 2, 3, and 4 will come up turned over. If after that he turns skewer number 1, then skewers number 1, 3, and 4 will be turned over, while skewer number 2 will be in the initial position, because it is the neighbor of skewer 1.

As we said before, the art of cooking requires perfect timing, so Miroslav wants to turn over all n skewers with the minimal possible number of actions. For example, for the above example $n = 6$, two turnings are sufficient: he can turn over skewers number 2 and 5.

Help Miroslav turn over all n skewers.

Input

The first line contains integer n ($1 \le n \le 50$) — the number of skewers on Miroslav's mangal.

Output

The first line should contain integer k — the minimal number of actions needed by Miroslav to turn over all skewers. Each of the next k lines should contain an integer from 1 to n denoting the number of the skewer that is to be turned over at the corresponding step.

Examples

standard input	standard output
6	2 2 5
5	2 2 5

Note

Note that in the second example it would also be correct to turn over skewers number 1 and 4, but it's incorrect to turn skewers number 2 and 4, because that way skewer number 3 will be turned to initial position.

Scoring

This problem has 50 tests checking all possible values of n in some arbitrary order. Each test **including samples** costs 2 points and is scored independently.

Problem B. Problem Development

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Innokentiy used to take part in programming competitions as a participant during his time in high school. Now, he is a student and takes part as a jury member and this time he is responsible for developing one of the competition problems. Here is its statement.

There are two bus stops denoted A and B, and there n buses that go from A to B every day. The shortest path from A to B takes t units of time but some buses might take longer paths. Moreover, buses are allowed to overtake each other during the route.

At each station one can find a sorted list of moments of time when a bus is at this station. We denote this list as $a_1 < a_2 < \ldots < a_n$ for stop A and as $b_1 < b_2 < \ldots < b_n$ for stop B. Denote as $p_1, p_2, \ldots, p_n$ some permutation of integers from 1 to n, i.e. such sequence that each integer from 1 to n inclusive occurs there exactly once. This permutation defines a correspondence between buses leaving stop A and buses arriving to stop B. That is, bus leaving stop A at moment a_1 arrives to stop B at moment b_{p_1}, bus leaving stop A at moment a_2 arrives to stop B at moment b_{p_2} and so on. We call correspondence p_i valid if time condition holds for each of the bus, i.e. $a_i + t \le b_{p_i}$. Moreover, it is guaranteed that identical permutation ($p_i = i$) defines a valid bus correspondence.

Then, the problem Innokentiy is developing asks to find out how late can each of the buses arrive to stop B, that is, for each i independently compute maximum possible value of $x_i \le n$, such that there exists a valid order where the i-th bus leaving stop A is the x_i-th bus to arrive to stop B, i.e. $p_i = x_i$.

Unfortunately, developing a programming contest problem includes developing its test set as well and now Innokentiy faces inverse problem. Given the number of buses n, time schedule for stop A a_i, parameter t and values of function x_i he has to find out any suitable time schedule for stop B b_i. That means that quadruple n, a_i, b_i and t defines a valid input for original problem with x_i as the answer.

Input

The first line of the input contains two integers n and t ($1 \le n \le 200\,000$, $1 \le t \le 10^{18}$) — the number of buses in time schedule for and the minimum possible travel time from stop A to stop B.

The second line contains n integers a_i ($1 \le a_1 < a_2 < \ldots < a_n \le 10^{18}$), defining the moments of time when bus leaves stop A.

The third line contains n integers x_i ($1 \le x_i \le n$), i-th of them stands for the maximum possible position, at which the i-th bus leaving stop A can arrive at stop B.

Output

If a solution exists, print "`Yes`" (without quotes) in the first line of the output. In the second line print n integers b_i ($1 \le b_1 < b_2 < \ldots < b_n \le 3 \cdot 10^{18}$). It is guaranteed that if there exists any solution, there exists a solution that satisfies b_i constraints. If there are multiple valid answer you can print any of them.

If there is no valid set of b_i print "`No`" (without quotes) in the only line of the output.

Examples

standard input	standard output

`3 10`
`4 6 8`
`2 2 3`
	`Yes`
	`16 17 21`

`2 1`
`1 2`
`2 1`
	`No`

Note

Scoring

Tests for this problem are divided into four groups. For each of the groups you earn points only if your solution passes all tests in this group and all tests in all of the **previous** groups.

Group	Points	Additional constraints			Comment
		n	t	a_i	
0	0	–	–	–	Sample tests.
1	10	$n = 2$	$t \le 10^9$	$a_i \le 10^9$	
2	20	$n = 3$	$t \le 10^9$	$a_i \le 10^9$	
3	40	$n \le 5000$	$t \le 10^9$	$a_i \le 10^9$	
4	30	–	–	–	

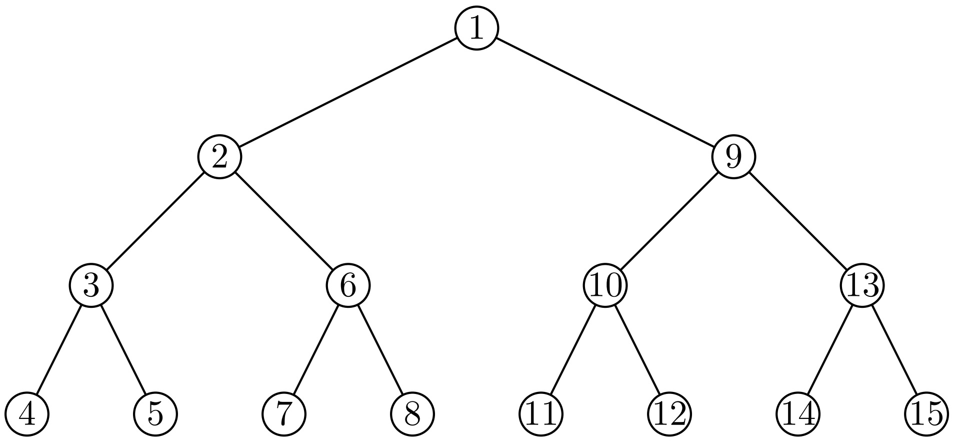
Problem C. Binary Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Complete binary tree of order 1 is a rooted tree consisting of a single node. Complete binary tree of order $k > 1$ is a rooted tree consisting of a node called root and two its children (denoted as left and right) that are roots of a complete binary trees of order $k - 1$.

Consider the following algorithm of enumerating nodes of a compete binary tree. Assign minimum unused positive integer to the root of the tree, then recursively run this algorithm for the left subtree of the tree, then recursively run the algorithm for the right subtree.

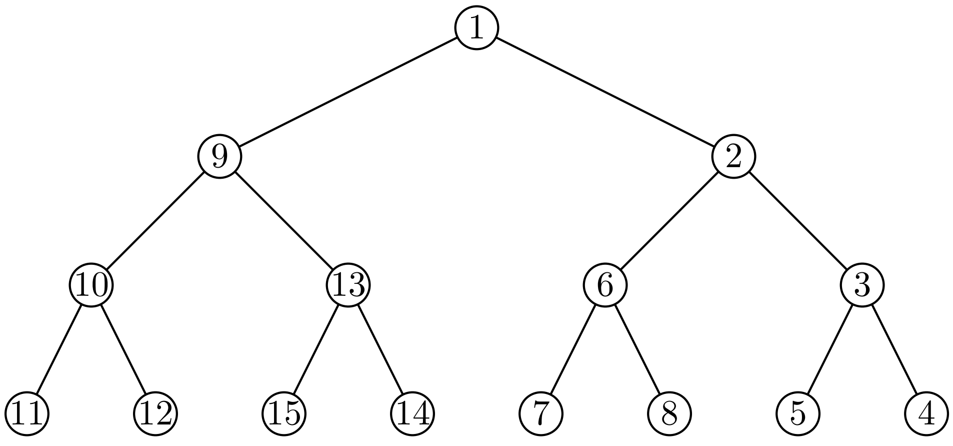
The picture below illustrates the resulting node enumeration for a complete binary tree of order 4.



Consider a complete binary tree of order k with n nodes of this tree being selected. The i-th selected node is assigned number a_i by applying the algorithm defined above to the tree. You are allowed to perform some (possibly none) operations on this tree of the following kind.

For some node v of the tree change the order of its children. After this operation the left child of this node becomes the right child and the right child becomes the left one.

For example, in the following tree these operations were performed on nodes 1, 2, 3 and 13.



After all operations are performed, nodes of the tree are enumerated again. For each of the selected nodes we have a special requirement that asks the i-th selected node a_i to take position b_i after re-numeration.

Determine if it's possible to swap children of some nodes in a way that after re-numeration all requirements are satisfied simultaneously.

Input

The first line contains two integers n and k ($1 \le n \le \min(300\,000, 2^k - 1)$, $1 \le k \le 60$), number of selected nodes and order of the given complete binary tree, respectively.

Next n lines contain pairs of integers a_i and b_i ($1 \le a_i, b_i \le 2^k - 1$), the i-th pair stands for the initial and the required assigned number of the i-th selected node.

It is guaranteed, that all a_i's are different and all b_i's are different.

Output

If there is a set of operations that makes all n selected nodes to have required numbers assigned to them after re-numeration (node which was initially a_i is assigned b_i for all i), output «`Yes`» (without quotes), otherwise output «`No`» (without quotes).

Examples

standard input	standard output

3 3
2 5
7 3
1 1
	Yes

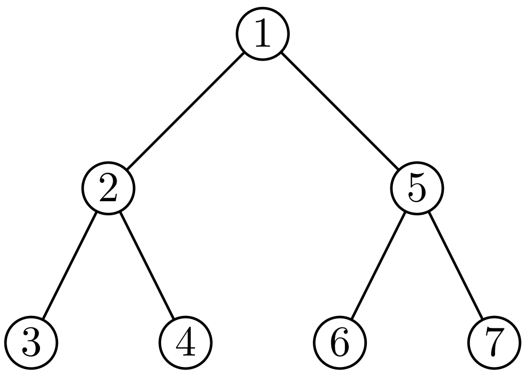
2 3
2 5
3 4
	No

1 2 1 3	No

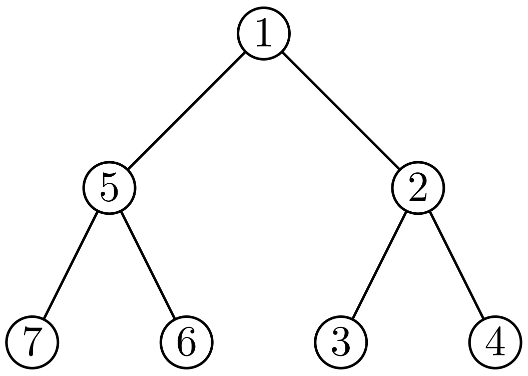
6 4
4 15
15 7
7 11
8 12
6 10
10 3
	Yes

Note

In the first example, the initial enumeration of tree nodes looks as follows:



One of the possible ways to achieve required enumeration is to perform described operations at nodes 1 and 5.



In the second example, one needs to change the children order at the root to make node 2 have number 5 after re-numeration. However this implies that node 3 can have only numbers 6 and 7 assigned to it after re-numeration.

In the third example, one can perform operation only at the root but no operation can changes number assigned to the root.

One of the possible solution to achieve required enumeration in the fourth example is shown in the problem statement.

Scoring

Tests for this problem are divided into seven groups. For each of the groups you earn points only if your solution passes all tests in this group and all tests in required groups.

Group	Points	Additional constraints		Required groups	Comment
		n	k		
0	0	–	–	–	Sample tests
1	10	$n \leq 31$	$k \leq 5$	0	–
2	20	–	$k \leq 20$	$0-1$	–
3	10	$n = 1$	–	–	–
4	10	$n \leq 2$	–	3	–
5	15	$n \leq 5000$	–	$0-1, 3-4$	–
6	25	$n \leq 50\,000$	–	$0-1, 3-5$	–
7	10	–	–	$0-6$	–

Problem D. Summer Oenothera Exhibition

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	512 megabytes

While some people enjoy spending their time solving programming contests, Dina prefers taking beautiful pictures. As soon as Byteland Botanical Garden announced Summer Oenothera Exhibition she decided to test her new camera there.

The exhibition consists of $l = 10^{100}$ Oenothera species arranged in a row and consecutively numbered with integers from 0 to $l - 1$. Camera lens allows to take a photo of w species on it, i.e. Dina can take a photo containing flowers with indices from x to $x + w - 1$ for some integer x between 0 and $l - w$. We will denote such photo with $[x, x + w - 1]$.

She has taken n photos, the i-th of which (in chronological order) is $[x_i, x_i + w - 1]$ in our notation. She decided to build a time-lapse video from these photos once she discovered that Oenothera blossoms open in the evening.

Dina takes each photo and truncates it, leaving its segment containing exactly k flowers, then she composes a video of these photos keeping their original order and voilà, a beautiful artwork has been created!

A scene is a contiguous sequence of photos such that the set of flowers on them is the same. The change between two scenes is called a *cut*. For example, consider the first photo contains flowers $[1, 5]$, the second photo contains flowers $[3, 7]$ and the third photo contains flowers $[8, 12]$. If $k = 3$, then Dina can truncate the first and the second photo into $[3, 5]$, and the third photo into $[9, 11]$. First two photos form a scene, third photo also forms a scene and the transition between these two scenes which happens between the second and the third photos is a cut. If $k = 4$, then each of the transitions between photos has to be a cut.

Dina wants the number of cuts to be as small as possible. Please help her! Calculate the minimum possible number of cuts for different values of k.

Input

The first line contains three positive integer n, w, q ($1 \le n, q \le 100\,000$, $1 \le w \le 10^9$) — the number of taken photos, the number of flowers on a single photo and the number of queries.

Next line contains n non-negative integers x_i ($0 \le x_i \le 10^9$) — the indices of the leftmost flowers on each of the photos.

Next line contains q positive integers k_i ($1 \le k_i \le w$) — the values of k for which you have to solve the problem.

It's guaranteed that all k_i are distinct.

Output

Print q integers — for each width of the truncated photo k_i, the minimum number of cuts that is possible.

Examples

standard input	standard output

3 6 5
2 4 0
1 2 3 4 5
	0
	0
	1
	1
	2

6 4 3
1 2 3 4 3 2
1 2 3
	0
	1
	2

Note

Scoring

Tests for this problem are divided into four groups. For each group you earn points only if your solution passes all tests in this group and all tests in all **required** groups except the sample tests.

Group	Tests	Point	Constraints		Required groups	Comment
			n	q		
0	1 – 2	0	–	–	–	Sample tests
1	3 – 17	15	$n \le 50$	$q \le 50$	–	$\lvert x_i - x_{i-1}\rvert \le 1$
2	18 – 55	20	$n \le 10\,000$	$q \le 10\,000$	1	
3	56 – 72	25	$n \le 100\,000$	$q \le 100\,000$	1	$\lvert x_i - x_{i-1}\rvert \le 1$
4	73 – 101	40	$n \le 100\,000$	$q \le 100\,000$	1, 2, 3	