Codeforces Round #409 (rated, Div. 1, based on VK Cup 2017 Round 2)

A. Voltage Keepsake

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You have n devices that you want to use simultaneously.

The i-th device uses a_i units of power per second. This usage is continuous. That is, in λ seconds, the device will use $\lambda \cdot a_i$ units of power. The i-th device currently has b_i units of power stored. All devices can store an arbitrary amount of power.

You have a single charger that can plug to any single device. The charger will add p units of power per second to a device. This charging is continuous. That is, if you plug in a device for λ seconds, it will gain $\lambda \cdot p$ units of power. You can switch which device is charging at any arbitrary unit of time (including real numbers), and the time it takes to switch is negligible.

You are wondering, what is the maximum amount of time you can use the devices until one of them hits 0 units of power.

If you can use the devices indefinitely, print `-1`. Otherwise, print the maximum amount of time before any one device hits 0 power.

Input

The first line contains two integers, n and p ($1 \le n \le 100\,000$, $1 \le p \le 10^9$) — the number of devices and the power of the charger.

This is followed by n lines which contain two integers each. Line i contains the integers a_i and b_i ($1 \le a_i, b_i \le 100\,000$) — the power of the device and the amount of power stored in the device in the beginning.

Output

If you can use the devices indefinitely, print `-1`. Otherwise, print the maximum amount of time before any one device hits 0 power.

Your answer will be considered correct if its absolute or relative error does not exceed 10^{-4}.

Namely, let's assume that your answer is a and the answer of the jury is b. The checker program will consider your answer correct if .

Examples

input
2 1 2 2 2 1000
output
2.0000000000

input
1 100 1 1
output
-1

input
3 5 4 3 5 2 6 1
output
0.5000000000

Note

In sample test 1, you can charge the first device for the entire time until it hits zero power. The second device has enough power to last this time without being charged.

In sample test 2, you can use the device indefinitely.

In sample test 3, we can charge the third device for $2 / 5$ of a second, then switch to charge the second device for a $1 / 10$ of a second.

B. Volatile Kite

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a convex polygon P with n distinct vertices $p_1, p_2, ..., p_n$. Vertex p_i has coordinates (x_i, y_i) in the 2D plane. These vertices are listed in clockwise order.

You can choose a real number D and move each vertex of the polygon a distance of at most D from their original positions.

Find the maximum value of D such that no matter how you move the vertices, the polygon does not intersect itself and stays convex.

Input

The first line has one integer n ($4 \le n \le 1\,000$) — the number of vertices.

The next n lines contain the coordinates of the vertices. Line i contains two integers x_i and y_i ($-10^9 \le x_i, y_i \le 10^9$) — the coordinates of the i-th vertex. These points are guaranteed to be given in clockwise order, and will form a strictly convex polygon (in particular, no three consecutive points lie on the same straight line).

Output

Print one real number D, which is the maximum real number such that no matter how you move the vertices, the polygon stays convex.

Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Namely, let's assume that your answer is a and the answer of the jury is b. The checker program will consider your answer correct if .

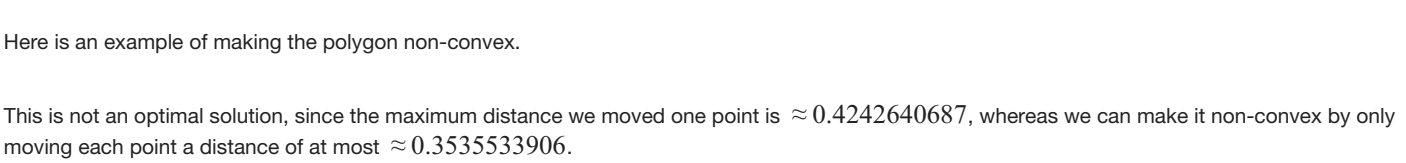
Examples

input
4 0 0 0 1 1 1 1 0
output
0.3535533906

input
6 5 0 10 0 12 -4 10 -8 5 -8 3 -4
output
1.0000000000

Note

Here is a picture of the first sample



Here is an example of making the polygon non-convex.

This is not an optimal solution, since the maximum distance we moved one point is ≈ 0.4242640687, whereas we can make it non-convex by only moving each point a distance of at most ≈ 0.3535533906.

C. Vulnerable Kerbals

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an integer m, and a list of n distinct integers between 0 and m - 1.

You would like to construct a sequence satisfying the properties:

- Each element is an integer between 0 and m - 1, inclusive.
- All prefix products of the sequence modulo m are distinct.
- No prefix product modulo m appears as an element of the input list.
- The length of the sequence is maximized.

Construct any sequence satisfying the properties above.

Input

The first line of input contains two integers n and m ($0 \leq n < m \leq 200\,000$) — the number of forbidden prefix products and the modulus.

If n is non-zero, the next line of input contains n distinct integers between 0 and m - 1, the forbidden prefix products. If n is zero, this line doesn't exist.

Output

On the first line, print the number k, denoting the length of your sequence.

On the second line, print k space separated integers, denoting your sequence.

Examples

input
0 5
output
5 1 2 4 3 0

input
3 10 2 9 1
output
6 3 9 2 9 8 0

Note

For the first case, the prefix products of this sequence modulo m are $[1, 2, 3, 4, 0]$.

For the second case, the prefix products of this sequence modulo m are $[3, 7, 4, 6, 8, 0]$.

D. Varying Kibibits

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given n integers $a_1, a_2, ..., a_n$. Denote this list of integers as T.

Let $f(L)$ be a function that takes in a non-empty list of integers L.

The function will output another integer as follows:

- First, all integers in L are padded with leading zeros so they are all the same length as the maximum length number in L.
- We will construct a string where the i-th character is the minimum of the i-th character in padded input numbers.
- The output is the number representing the string interpreted in base 10.

For example $f(10, 9) = 0, f(123, 321) = 121, f(530, 932, 81) = 30$.

Define the function

Here, denotes a subsequence.
In other words, $G(x)$ is the sum of squares of sum of elements of nonempty subsequences of T that evaluate to x when plugged into f modulo $1\,000\,000\,007$, then multiplied by x. The last multiplication is not modded.

You would like to compute $G(0), G(1), ..., G(999\,999)$. To reduce the output size, print the value , where denotes the bitwise XOR operator.

Input

The first line contains the integer n $(1 \le n \le 1\,000\,000)$ — the size of list T.

The next line contains n space-separated integers, $a_1, a_2, ..., a_n$ $(0 \le a_i \le 999\,999)$ — the elements of the list.

Output

Output a single integer, the answer to the problem.

Examples

input
3 123 321 555
output
292711924

input
1 999999
output
997992010006992

input
10 1 1 1 1 1 1 1 1 1 1
output
28160

Note

For the first sample, the nonzero values of G are $G(121) = 144\,611\,577$, $G(123) = 58\,401\,999$, $G(321) = 279\,403\,857$, $G(555) = 170\,953\,875$. The bitwise XOR of these numbers is equal to $292\,711\,924$.

For example, , since the subsequences $[123]$ and $[123, 555]$ evaluate to 123 when plugged into f.

For the second sample, we have

For the last sample, we have , where is the binomial coefficient.

E. Verifying Kingdom

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is an interactive problem.

The judge has a hidden rooted full binary tree with n leaves. A full binary tree is one where every node has either 0 or 2 children. The nodes with 0 children are called the leaves of the tree. Since this is a full binary tree, there are exactly $2n$ - 1 nodes in the tree. The leaves of the judge's tree has labels from 1 to n. You would like to reconstruct a tree that is isomorphic to the judge's tree. To do this, you can ask some questions.

A question consists of printing the label of three distinct leaves a_1, a_2, a_3. Let the *depth* of a node be the shortest distance from the node to the root of the tree. Let $LCA(a, b)$ denote the node with maximum depth that is a common ancestor of the nodes a and b.

Consider $X = LCA(a_1, a_2), Y = LCA(a_2, a_3), Z = LCA(a_3, a_1)$. The judge will tell you which one of X, Y, Z has the maximum depth. Note, this pair is uniquely determined since the tree is a binary tree; there can't be any ties.

More specifically, if X (or Y, Z respectively) maximizes the depth, the judge will respond with the string "`X`" (or "`Y`", "`Z`" respectively).

You may only ask at most $10 \cdot n$ questions.

Input

The first line of input will contain a single integer n ($3 \le n \le 1\,000$) — the number of leaves in the tree.

Output

To print the final answer, print out the string "`-1`" on its own line. Then, the next line should contain $2n$ - 1 integers. The i-th integer should be the parent of the i-th node, or -1, if it is the root.

Your answer will be judged correct if your output is isomorphic to the judge's tree. In particular, the labels of the leaves do not need to be labeled from 1 to n. Here, isomorphic means that there exists a permutation π such that node i is the parent of node j in the judge tree if and only node $\pi(i)$ is the parent of node $\pi(j)$ in your tree.

Interaction

To ask a question, print out three distinct integers a_1, a_2, a_3. These integers should be between 1 and n, inclusive.

The judge will respond with a single character, either "`X`", "`Y`", "`Z`".

If the string is "`X`" (or "`Y`", "`Z`" respectively), that means the pair (a_1, a_2) (or (a_2, a_3), (a_3, a_1) respectively) has the deepest LCA among the three pairs.

You may only ask a question at most $10 \cdot n$ times, otherwise, you will get `Wrong Answer`.

When you are ready to answer, print out a single integer "`-1`" on its own line. The next line should contain $2n$ - 1 integers. The i-th integer should be the parent of the i-th node, or -1, if it is the root. Do not forget to flush the final answer as well. Printing the answer does not count as asking a question.

You will get `Wrong Answer` verdict if

- Your question or answers are not in the format described in this statement.
- You ask strictly more than $10 \cdot n$ questions.
- Your question contains duplicate indices.
- Your final answer is not isomorphic to the judge tree.

You will get `Idleness Limit Exceeded` if you don't print anything or if you forget to flush the output, including for the final answer (more info about flushing output below).

To flush you can use (just after printing an integer and end-of-line):

- `fflush(stdout)` in C++;
- `System.out.flush()` in Java;
- `stdout.flush()` in Python;
- `flush(output)` in Pascal;
- See the documentation for other languages.

If at any moment your program reads `-1` as an answer, it should immediately exit normally (for example, by calling `exit(0)`). You will get `Wrong Answer` in this case, it means that you made more queries than allowed, or made an invalid query. If you ignore this, you can get other verdicts since your program will continue to read from a closed stream.

Hacking To hack someone, use the following format

n

`p_1 p_2 ... p_{2n-1}`

This denotes a tree where the parent of the i-th node is p_i ($p_i = -1$ or $n < p_i \le 2n - 1$). If p_i is equal to `-1`, then node i is the root. This input must describe a valid full rooted binary tree.

Of course, contestant programs will not be able to see this input.

Example

input
5 X Z Y Y X
output
1 4 2 1 2 4 2 4 1 2 3 5 2 4 3 -1 -1 1 1 2 2 3 3 6 6

Note

For the first sample, the judge has the hidden tree:

Here is a more readable format of the interaction:

The last line can also be `8 6 9 8 9 7 -1 6 7`.