Codeforces Round #563 (Div. 2)

A. Ehab Fails to Be Thanos

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You're given an array a of length $2n$. Is it possible to reorder it in such way so that the sum of the first n elements **isn't** equal to the sum of the last n elements?

Input
The first line contains an integer n ($1 \le n \le 1000$), where $2n$ is the number of elements in the array a.

The second line contains $2n$ space-separated integers $a_1, a_2, \ldots, a_{2n}$ ($1 \le a_i \le 10^6$) — the elements of the array a.

Output
If there's no solution, print "-1" (without quotes). Otherwise, print a single line containing $2n$ space-separated integers. They must form a reordering of a. You are allowed to not change the order.

Examples

input
3 1 2 2 1 3 1
output
2 1 3 1 1 2

input
1 1 1
output
-1

Note
In the first example, the first n elements have sum $2 + 1 + 3 = 6$ while the last n elements have sum $1 + 1 + 2 = 4$. The sums aren't equal.

In the second example, there's no solution.

B. Ehab Is an Odd Person

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You're given an array a of length n. You can perform the following operation on it as many times as you want:

- Pick two integers i and j $(1 \le i, j \le n)$ such that $a_i + a_j$ **is odd**, then swap a_i and a_j.

What is lexicographically the smallest array you can obtain?

An array x is lexicographically smaller than an array y if there exists an index i such that $x_i < y_i$, and $x_j = y_j$ for all $1 \le j < i$. Less formally, at the first index i in which they differ, $x_i < y_i$

Input
The first line contains an integer n ($1 \le n \le 10^5$) — the number of elements in the array a.

The second line contains n space-separated integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the elements of the array a.

Output
The only line contains n space-separated integers, the lexicographically smallest array you can obtain.

Examples

input
3 4 1 7

output
1 4 7

input
2 1 1
output
1 1

Note
In the first example, we can swap 1 and 4 since $1 + 4 = 5$, which is odd.

C. Ehab and a Special Coloring Problem

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You're given an integer n. For every integer i from 2 to n, assign a positive integer a_i such that the following conditions hold:

- For any pair of integers (i, j), if i and j are coprime, $a_i \neq a_j$.
- The maximal value of all a_i should be minimized (that is, as small as possible).

A pair of integers is called coprime if their greatest common divisor is 1.

Input
The only line contains the integer n ($2 \leq n \leq 10^5$).

Output
Print $n - 1$ integers, a_2, a_3, ..., a_n ($1 \leq a_i \leq n$).

If there are multiple solutions, print any of them.

Examples

input
4
output
1 2 1

input
3
output
2 1

Note
In the first example, notice that 3 and 4 are coprime, so $a_3 \neq a_4$. Also, notice that $a = [1, 2, 3]$ satisfies the first condition, but it's not a correct answer because its maximal value is 3.

D. Ehab and the Expected XOR Problem

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Given two integers n and x, construct an array that satisfies the following conditions:

- for any element a_i in the array, $1 \leq a_i < 2^n$;
- there is no **non-empty** subsegment with bitwise XOR equal to 0 or x,
- its length l should be maximized.

A sequence b is a subsegment of a sequence a if b can be obtained from a by deletion of several (possibly, zero or all) elements from the beginning and several (possibly, zero or all) elements from the end.

Input
The only line contains two integers n and x ($1 \leq n \leq 18$, $1 \leq x < 2^{18}$).

Output
The first line should contain the length of the array l.

If l is positive, the second line should contain l space-separated integers a_1, a_2, ..., a_l ($1 \le a_i < 2^n$) — the elements of the array a.

If there are multiple solutions, print any of them.

Examples

input
3 5
output
3 6 1 3

input
2 4
output
3 1 3 1

input
1 1
output
0

Note
In the first example, the bitwise XOR of the subsegments are $\{6, 7, 4, 1, 2, 3\}$.

E. Ehab and the Expected GCD Problem

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

Let's define a function $f(p)$ on a permutation p as follows. Let g_i be the greatest common divisor (GCD) of elements p_1, p_2, ..., p_i (in other words, it is the GCD of the prefix of length i). Then $f(p)$ is the number of **distinct** elements among g_1, g_2, ..., g_n.

Let $f_{max}(n)$ be the maximum value of $f(p)$ among all permutations p of integers 1, 2, ..., n.

Given an integers n, count the number of permutations p of integers 1, 2, ..., n, such that $f(p)$ is equal to $f_{max}(n)$. Since the answer may be large, print the remainder of its division by $1000\,000\,007 = 10^9 + 7$.

Input
The only line contains the integer n ($2 \le n \le 10^6$) — the length of the permutations.

Output
The only line should contain your answer modulo $10^9 + 7$.

Examples

input
2
output
1

input
3
output
4

input
6
output
120

Note
Consider the second example: these are the permutations of length 3:

- $[1, 2, 3]$, $f(p) = 1$.
- $[1, 3, 2]$, $f(p) = 1$.

- $[2, 1, 3]$, $f(p) = 2$.
- $[2, 3, 1]$, $f(p) = 2$.
- $[3, 1, 2]$, $f(p) = 2$.
- $[3, 2, 1]$, $f(p) = 2$.

The maximum value $f_{max}(3) = 2$, and there are 4 permutations p such that $f(p) = 2$.

F. Ehab and the Big Finale

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

This is an interactive problem.

You're given a tree consisting of n nodes, rooted at node 1. A tree is a connected graph with no cycles.

We chose a hidden node x. In order to find this node, you can ask queries of two types:

- d u ($1 \le u \le n$). We will answer with the distance between nodes u and x. The distance between two nodes is the number of edges in the shortest path between them.
- s u ($1 \le u \le n$). We will answer with the second node on the path from u to x. However, there's a plot twist. If u is **not** an ancestor of x, you'll receive "`Wrong answer`" verdict!

Node a is called an ancestor of node b if $a \neq b$ and the shortest path from node 1 to node b passes through node a. **Note that in this problem a node is not an ancestor of itself**.

Can you find x in no more than 36 queries? The hidden node is fixed in each test beforehand and does not depend on your queries.

Input

The first line contains the integer n ($2 \le n \le 2 \cdot 10^5$) — the number of nodes in the tree.

Each of the next $n - 1$ lines contains two space-separated integers u and v ($1 \le u, v \le n$) that mean there's an edge between nodes u and v. It's guaranteed that the given graph is a tree.

Output

To print the answer, print "`! x`" (without quotes).

Interaction

To ask a question, print it in one of the formats above:

- d u ($1 \le u \le n$), or
- s u ($1 \le u \le n$).

After each question, you should read the answer: either the distance or the second vertex on the path, as mentioned in the legend.

If we answer with -1 instead of a valid answer, that means you exceeded the number of queries, made an invalid query, or violated the condition in the second type of queries. Exit immediately after receiving -1 and you will see `Wrong answer` verdict. Otherwise, you can get an arbitrary verdict because your solution will continue to read from a closed stream.

After printing a query, do not forget to output end of line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

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

Hacks:

The first line should contain two integers n and x ($2 \le n \le 2 \cdot 10^5$, $1 \le x \le n$).

Each of the next $n - 1$ lines should contain two integers u and v ($1 \le u, v \le n$) that mean there is an edge between nodes u and v. The edges must form a tree.

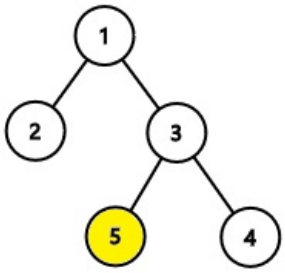
Example

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

```
d 2
s 3
! 5
```

Note

In the first example, the hidden node is node 5.



We first ask about the distance between node x and node 2. The answer is 3, so node x is either 4 or 5. We then ask about the second node in the path from node 3 to node x. Note here that node 3 is an ancestor of node 5. We receive node 5 as the answer. Finally, we report that the hidden node is node 5.
