Codeforces Round #554 (Div. 2)

A. Neko Finds Grapes

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

On a random day, Neko found n treasure chests and m keys. The i-th chest has an integer a_i written on it and the j-th key has an integer b_j on it. Neko knows those chests contain the powerful mysterious green Grapes, thus Neko wants to open as many treasure chests as possible.

The j-th key can be used to unlock the i-th chest if and only if the sum of the key number and the chest number is an odd number. Formally, $a_i + b_j \equiv 1 \pmod 2$. One key can be used to open at most one chest, and one chest can be opened at most once.

Find the maximum number of chests Neko can open.

Input

The first line contains integers n and m ($1 \le n, m \le 10^5$) — the number of chests and the number of keys.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the numbers written on the treasure chests.

The third line contains m integers $b_1, b_2, \ldots, b_m$ ($1 \le b_i \le 10^9$) — the numbers written on the keys.

Output

Print the maximum number of chests you can open.

Examples

input
5 4 9 14 6 2 11 8 4 7 20
output
3

input
5 1 2 4 6 8 10 5
output
1

input
1 4 10 20 30 40 50
output
0

Note

In the first example, one possible way to unlock 3 chests is as follows:

- Use first key to unlock the fifth chest,
- Use third key to unlock the second chest,
- Use fourth key to unlock the first chest.

In the second example, you can use the only key to unlock any single chest (note that one key can't be used twice).

In the third example, no key can unlock the given chest.

B. Neko Performs Cat Furrier Transform

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

Cat Furrier Transform is a popular algorithm among cat programmers to create longcats. As one of the greatest cat programmers ever exist, Neko wants to utilize this algorithm to create the perfect longcat.

Assume that we have a cat with a number x. A perfect longcat is a cat with a number equal $2^m - 1$ for some non-negative integer m. For example, the numbers 0, 1, 3, 7, 15 and so on are suitable for the perfect longcats.

In the Cat Furrier Transform, the following operations can be performed on x:

- (Operation A): you select any non-negative integer n and replace x with $x \oplus (2^n - 1)$, with $\oplus$ being a bitwise XOR operator.
- (Operation B): replace x with $x + 1$.

The first applied operation must be of type A, the second of type B, the third of type A again, and so on. Formally, if we number operations from one in the order they are executed, then odd-numbered operations must be of type A and the even-numbered operations must be of type B.

Neko wants to produce perfect longcats at industrial scale, thus for each cat Neko only wants to perform at most 40 operations. Can you help Neko writing a transformation plan?

Note that it is **not required** to minimize the number of operations. You just need to use no more than 40 operations.

Input
The only line contains a single integer x ($1 \le x \le 10^6$).

Output
The first line should contain a single integer t ($0 \le t \le 40$) — the number of operations to apply.

Then for each odd-numbered operation print the corresponding number n_i in it. That is, print $\lceil \frac{t}{2} \rceil$ integers n_i ($0 \le n_i \le 30$), denoting the replacement x with $x \oplus (2^{n_i} - 1)$ in the corresponding step.

If there are multiple possible answers, you can print any of them. It is possible to show, that there is at least one answer in the constraints of this problem.

Examples

input
39
output
4 5 3

input
1
output
0

input
7
output
0

Note
In the first test, one of the transforms might be as follows: $39 \to 56 \to 57 \to 62 \to 63$. Or more precisely:

1. Pick $n = 5$. x is transformed into $39 \oplus 31$, or 56.
2. Increase x by 1, changing its value to 57.
3. Pick $n = 3$. x is transformed into $57 \oplus 7$, or 62.
4. Increase x by 1, changing its value to $63 = 2^6 - 1$.

In the second and third test, the number already satisfies the goal requirement.

C. Neko does Maths

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

Neko loves divisors. During the latest number theory lesson, he got an interesting exercise from his math teacher.

Neko has two integers a and b. His goal is to find a non-negative integer k such that the least common multiple of $a + k$ and $b + k$ is the smallest possible. If there are multiple optimal integers k, he needs to choose the smallest one.

Given his mathematical talent, Neko had no trouble getting Wrong Answer on this problem. Can you help him solve it?

Input
The only line contains two integers a and b ($1 \le a, b \le 10^9$).

Output
Print the smallest non-negative integer k ($k \ge 0$) such that the lowest common multiple of $a + k$ and $b + k$ is the smallest possible.

If there are many possible integers k giving the same value of the least common multiple, print the smallest one.

Examples

input
6 10
output
2

input
21 31
output
9

input
5 10
output
0

Note
In the first test, one should choose $k = 2$, as the least common multiple of $6 + 2$ and $10 + 2$ is 24, which is the smallest least common multiple possible.

D. Neko and Aki's Prank

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

Neko is playing with his toys on the backyard of Aki's house. Aki decided to play a prank on him, by secretly putting catnip into Neko's toys. Unfortunately, he went overboard and put an entire bag of catnip into the toys...

It took Neko an entire day to turn back to normal. Neko reported to Aki that he saw a lot of weird things, including a trie of all correct bracket sequences of length $2n$.

The definition of correct bracket sequence is as follows:

- The empty sequence is a correct bracket sequence,
- If s is a correct bracket sequence, then $(\, s \,)$ is a correct bracket sequence,
- If s and t are a correct bracket sequence, then st is also a correct bracket sequence.

For example, the strings "(( ))", "( ) ( )" form a correct bracket sequence, while ") (" and "((" not.

Aki then came up with an interesting problem: What is the size of the maximum matching (the largest set of edges such that there are no two edges with a common vertex) in this trie? Since the answer can be quite large, print it modulo $10^9 + 7$.

Input
The only line contains a single integer n ($1 \le n \le 1000$).

Output
Print exactly one integer — the size of the maximum matching in the trie. Since the answer can be quite large, print it modulo $10^9 + 7$.

Examples

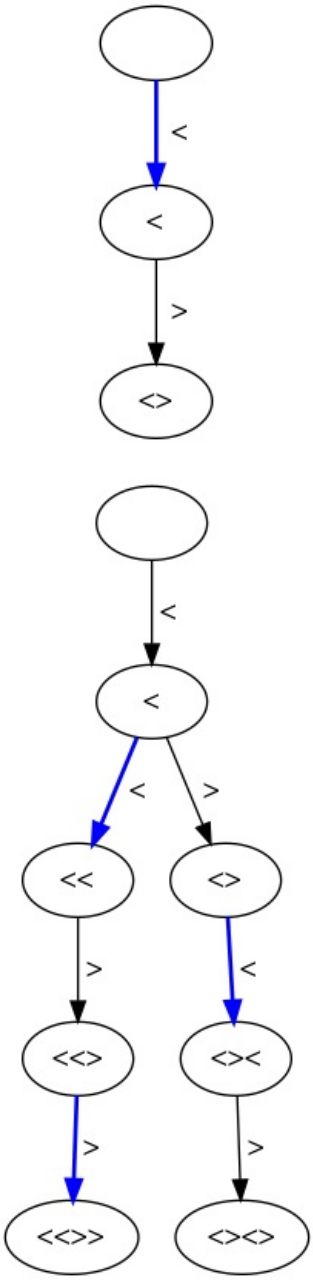
input
1
output
1

input
2
output
3

input
3
output
9

Note

The pictures below illustrate tries in the first two examples (for clarity, the round brackets are replaced with angle brackets). The maximum matching is highlighted with blue.



E. Neko and Flashback

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

A *permutation* of length k is a sequence of k integers from 1 to k containing each integer exactly once. For example, the sequence $[3,1,2]$ is a permutation of length 3.

When Neko was five, he thought of an array a of n positive integers and a permutation p of length $n-1$. Then, he performed the following:

- Constructed an array b of length $n-1$, where $b_i = \min(a_i, a_{i+1})$.
- Constructed an array c of length $n-1$, where $c_i = \max(a_i, a_{i+1})$.
- Constructed an array b' of length $n-1$, where $b'_i = b_{p_i}$.
- Constructed an array c' of length $n-1$, where $c'_i = c_{p_i}$.

For example, if the array a was $[3,4,6,5,7]$ and permutation p was $[2,4,1,3]$, then Neko would have constructed the following

arrays:

- $b = [3, 4, 5, 5]$
- $c = [4, 6, 6, 7]$
- $b' = [4, 5, 3, 5]$
- $c' = [6, 7, 4, 6]$

Then, he wrote two arrays b' and c' on a piece of paper and forgot about it. 14 years later, when he was cleaning up his room, he discovered this old piece of paper with two arrays b' and c' written on it. However he can't remember the array a and permutation p he used.

In case Neko made a mistake and there is no array a and permutation p resulting in such b' and c', print -1. Otherwise, help him recover any possible array a.

Input

The first line contains an integer n ($2 \leq n \leq 10^5$) — the number of elements in array a.

The second line contains $n - 1$ integers $b'_1, b'_2, \ldots, b'_{n-1}$ ($1 \leq b'_i \leq 10^9$).

The third line contains $n - 1$ integers $c'_1, c'_2, \ldots, c'_{n-1}$ ($1 \leq c'_i \leq 10^9$).

Output

If Neko made a mistake and there is no array a and a permutation p leading to the b' and c', print -1. Otherwise, print n positive integers a_i ($1 \leq a_i \leq 10^9$), denoting the elements of the array a.

If there are multiple possible solutions, print any of them.

Examples

input
5 4 5 3 5 6 7 4 6
output
3 4 6 5 7

input
3 2 4 3 2
output
-1

input
8 2 3 1 1 2 4 3 3 4 4 2 5 5 4
output
3 4 5 2 1 4 3 2

Note

The first example is explained is the problem statement.

In the third example, for $a = [3, 4, 5, 2, 1, 4, 3, 2]$, a possible permutation p is $[7, 1, 5, 4, 3, 2, 6]$. In that case, Neko would have constructed the following arrays:

- $b = [3, 4, 2, 1, 1, 3, 2]$
- $c = [4, 5, 5, 2, 4, 4, 3]$
- $b' = [2, 3, 1, 1, 2, 4, 3]$
- $c' = [3, 4, 4, 2, 5, 5, 4]$

F1. Neko Rules the Catniverse (Small Version)

time limit per test: 7 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

This problem is same as the next one, but has smaller constraints.

Aki is playing a new video game. In the video game, he will control Neko, the giant cat, to fly between planets in the Catniverse.

There are n planets in the Catniverse, numbered from 1 to n. At the beginning of the game, Aki chooses the planet where Neko is initially located. Then Aki performs $k - 1$ moves, where in each move Neko is moved from the current planet x to some other planet

y such that:

- Planet y is not visited yet.
- $1 \le y \le x + m$ (where m is a fixed constant given in the input)

This way, Neko will visit exactly k different planets. Two ways of visiting planets are called different if there is some index i such that the i-th planet visited in the first way is different from the i-th planet visited in the second way.

What is the total number of ways to visit k planets this way? Since the answer can be quite large, print it modulo $10^9 + 7$.

Input

The only line contains three integers n, k and m ($1 \le n \le 10^5$, $1 \le k \le \min(n, 12)$, $1 \le m \le 4$) — the number of planets in the Catniverse, the number of planets Neko needs to visit and the said constant m.

Output

Print exactly one integer — the number of different ways Neko can visit exactly k planets. Since the answer can be quite large, print it modulo $10^9 + 7$.

Examples

input
3 3 1
output
4

input
4 2 1
output
9

input
5 5 4
output
120

input
100 1 2
output
100

Note

In the first example, there are 4 ways Neko can visit all the planets:

- $1 \to 2 \to 3$
- $2 \to 3 \to 1$
- $3 \to 1 \to 2$
- $3 \to 2 \to 1$

In the second example, there are 9 ways Neko can visit exactly 2 planets:

- $1 \to 2$
- $2 \to 1$
- $2 \to 3$
- $3 \to 1$
- $3 \to 2$
- $3 \to 4$
- $4 \to 1$
- $4 \to 2$
- $4 \to 3$

In the third example, with $m = 4$, Neko can visit all the planets in any order, so there are $5! = 120$ ways Neko can visit all the planets.

In the fourth example, Neko only visit exactly 1 planet (which is also the planet he initially located), and there are 100 ways to choose the starting planet for Neko.

F2. Neko Rules the Catniverse (Large Version)

time limit per test: 7 seconds
memory limit per test: 256 megabytes

input: standard input
output: standard output

This problem is same as the previous one, but has larger constraints.

Aki is playing a new video game. In the video game, he will control Neko, the giant cat, to fly between planets in the Catniverse.

There are n planets in the Catniverse, numbered from 1 to n. At the beginning of the game, Aki chooses the planet where Neko is initially located. Then Aki performs $k - 1$ moves, where in each move Neko is moved from the current planet x to some other planet y such that:

- Planet y is not visited yet.
- $1 \leq y \leq x + m$ (where m is a fixed constant given in the input)

This way, Neko will visit exactly k different planets. Two ways of visiting planets are called different if there is some index i such that, the i-th planet visited in the first way is different from the i-th planet visited in the second way.

What is the total number of ways to visit k planets this way? Since the answer can be quite large, print it modulo $10^9 + 7$.

Input
The only line contains three integers n, k and m ($1 \leq n \leq 10^9$, $1 \leq k \leq \min(n, 12)$, $1 \leq m \leq 4$) — the number of planets in the Catniverse, the number of planets Neko needs to visit and the said constant m.

Output
Print exactly one integer — the number of different ways Neko can visit exactly k planets. Since the answer can be quite large, print it modulo $10^9 + 7$.

Examples

input
3 3 1
output
4

input
4 2 1
output
9

input
5 5 4
output
120

input
100 1 2
output
100

Note
In the first example, there are 4 ways Neko can visit all the planets:

- $1 \to 2 \to 3$
- $2 \to 3 \to 1$
- $3 \to 1 \to 2$
- $3 \to 2 \to 1$

In the second example, there are 9 ways Neko can visit exactly 2 planets:

- $1 \to 2$
- $2 \to 1$
- $2 \to 3$
- $3 \to 1$
- $3 \to 2$
- $3 \to 4$
- $4 \to 1$
- $4 \to 2$
- $4 \to 3$

In the third example, with $m = 4$, Neko can visit all the planets in any order, so there are $5! = 120$ ways Neko can visit all the planets.

In the fourth example, Neko only visit exactly 1 planet (which is also the planet he initially located), and there are 100 ways to choose the starting planet for Neko.
