Codeforces Round #177 (Div. 2)

A. Polo the Penguin and Segments

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

Little penguin Polo adores integer segments, that is, pairs of integers $[l; r]$ $(l \le r)$.

He has a set that consists of n integer segments: $[l_1; r_1], [l_2; r_2], \ldots, [l_n; r_n]$. We know that no two segments of this set intersect. In one move Polo can either widen any segment of the set 1 unit to the left or 1 unit to the right, that is transform $[l; r]$ to either segment $[l - 1; r]$, or to segment $[l; r + 1]$.

The *value* of a set of segments that consists of n segments $[l_1; r_1], [l_2; r_2], \ldots, [l_n; r_n]$ is the number of integers x, such that there is integer j, for which the following inequality holds, $l_j \le x \le r_j$.

Find the minimum number of moves needed to make the value of the set of Polo's segments divisible by k.

Input

The first line contains two integers n and k $(1 \le n, k \le 10^5)$. Each of the following n lines contain a segment as a pair of integers l_i and r_i $(-10^5 \le l_i \le r_i \le 10^5)$, separated by a space.

It is guaranteed that no two segments intersect. In other words, for any two integers i, j $(1 \le i < j \le n)$ the following inequality holds, $min(r_i, r_j) < max(l_i, l_j)$.

Output

In a single line print a single integer — the answer to the problem.

Examples

input
2 3 1 2 3 4
output
2

input
3 7 1 2 3 3 4 7
output
0

B. Polo the Penguin and Matrix

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

Little penguin Polo has an $n \times m$ matrix, consisting of integers. Let's index the matrix rows from 1 to n from top to bottom and let's index the columns from 1 to m from left to right. Let's represent the matrix element on the intersection of row i and column j as a_{ij}.

In one move the penguin can add or subtract number d from some matrix element. Find the minimum number of moves needed to make all matrix elements equal. If the described plan is impossible to carry out, say so.

Input

The first line contains three integers n, m and d $(1 \le n, m \le 100, 1 \le d \le 10^4)$ — the matrix sizes and the d parameter. Next n lines contain the matrix: the j-th integer in the i-th row is the matrix element a_{ij} $(1 \le a_{ij} \le 10^4)$.

Output

In a single line print a single integer — the minimum number of moves the penguin needs to make all matrix elements equal. If that is impossible, print "-1" (without the quotes).

Examples

input
2 2 2 2 4 6 8
output
4

input
1 2 7 6 7
output
-1

C. Polo the Penguin and Strings

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

Little penguin Polo adores strings. But most of all he adores strings of length n.

One day he wanted to find a string that meets the following conditions:

1. The string consists of n lowercase English letters (that is, the string's length equals n), exactly k of these letters are distinct.
2. No two neighbouring letters of a string coincide; that is, if we represent a string as $s = s_1 s_2 \ldots s_n$, then the following inequality holds, $s_i \neq s_{i+1} (1 \leq i < n)$.
3. Among all strings that meet points 1 and 2, the required string is lexicographically smallest.

Help him find such string or state that such string doesn't exist.

String $x = x_1 x_2 \ldots x_p$ is *lexicographically less* than string $y = y_1 y_2 \ldots y_q$, if either $p < q$ and $x_1 = y_1, x_2 = y_2, \ldots, x_p = y_p$, or there is such number r $(r < p, r < q)$, that $x_1 = y_1, x_2 = y_2, \ldots, x_r = y_r$ and $x_{r+1} < y_{r+1}$. The characters of the strings are compared by their ASCII codes.

Input

A single line contains two positive integers n and k $(1 \leq n \leq 10^6, 1 \leq k \leq 26)$ — the string's length and the number of distinct letters.

Output

In a single line print the required string. If there isn't such string, print "-1" (without the quotes).

Examples

input
7 4
output
ababacd

input
4 7
output
-1

D. Polo the Penguin and Houses

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

Little penguin Polo loves his home village. The village has n houses, indexed by integers from 1 to n. Each house has a plaque containing an integer, the i-th house has a plaque containing integer p_i ($1 \le p_i \le n$).

Little penguin Polo loves walking around this village. The walk looks like that. First he stands by a house number x. Then he goes to the house whose number is written on the plaque of house x (that is, to house p_x), then he goes to the house whose number is written on the plaque of house p_x (that is, to house p_{p_x}), and so on.

We know that:

1. When the penguin starts walking from any house indexed from 1 to k, inclusive, he can walk to house number 1.
2. When the penguin starts walking from any house indexed from $k + 1$ to n, inclusive, he definitely cannot walk to house number 1.
3. When the penguin starts walking from house number 1, he can get back to house number 1 after some non-zero number of walks from a house to a house.

You need to find the number of ways you may write the numbers on the houses' plaques so as to fulfill the three above described conditions. Print the remainder after dividing this number by 1000000007 ($10^9 + 7$).

Input

The single line contains two space-separated integers n and k ($1 \le n \le 1000, 1 \le k \le min(8, n)$) — the number of the houses and the number k from the statement.

Output

In a single line print a single integer — the answer to the problem modulo 1000000007 ($10^9 + 7$).

Examples

input
5 2
output
54

input
7 4
output
1728

E. Polo the Penguin and XOR operation

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

Little penguin Polo likes permutations. But most of all he likes permutations of integers from 0 to n, inclusive.

For permutation $p = p_0, p_1, \ldots, p_n$, Polo has defined its beauty — number $(0 \oplus p_0) + (1 \oplus p_1) + \cdots + (n \oplus p_n)$.

Expression $x \oplus y$ means applying the operation of bitwise excluding "OR" to numbers x and y. This operation exists in all modern programming languages, for example, in language *C++* and *Java* it is represented as "^" and in *Pascal* — as "xor".

Help him find among all permutations of integers from 0 to n the permutation with the maximum beauty.

Input

The single line contains a positive integer n ($1 \le n \le 10^6$).

Output

In the first line print integer m the maximum possible beauty. In the second line print any permutation of integers from 0 to n with the beauty equal to m.

If there are several suitable permutations, you are allowed to print any of them.

Examples

input
4

output
20 0 2 1 4 3