A. 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 $s = s_1s_2... s_n$, then the following inequality holds, $s_i \neq s_{i+1}(1 \le i < n)$.
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 $x = x_1x_2... x_p$ is *lexicographically less* than string $y = y_1y_2... y_q$, if either $p < q$ and $x_1 = y_1, x_2 = y_2, ... , x_p = y_p$, or there is such number r $(r < p, r < q)$, that $x_1 = y_1, x_2 = y_2, ... , x_r = y_r$ 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 $(1 \le n \le 10^6, 1 \le k \le 26)$ — 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

```
abababacd
```

input

```
4 7
```

output

```
-1
```

B. 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

C. 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 $p = p_0, p_1, ..., p_n$, 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
```

D. Polo the Penguin and Trees

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

Little penguin Polo has got a tree — a non-directed connected acyclic graph, containing n nodes and $n - 1$ edges. We will consider the tree nodes numbered by integers from 1 to n.

Today Polo wonders, how to find the number of pairs of paths that don't have common nodes. More formally, he should find the number of groups of four integers a, b, c and d such that:

- $1 \le a < b \le n$;
- $1 \le c < d \le n$;
- there's no such node that lies on both the shortest path from node a to node b and from node c to node d.

The shortest path betweem two nodes is the path that is shortest in the number of edges.

Help Polo solve this problem.

Input

The first line contains integer n $(1 \le n \le 80000)$ — the number of tree nodes. Each of the following $n - 1$ lines contains a pair of integers u_i and v_i $(1 \le u_i, v_i \le n; u_i \ne v_i)$ — the i-th edge of the tree.

It is guaranteed that the given graph is a tree.

Output

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

Please do not use the %lld specificator to read or write 64-bit numbers in C++. It is recommended to use the cin, cout streams or the %I64d specificator.

Examples

input

```
4
1 2
2 3
3 4
```

output

```
2
```

E. Polo the Penguin and Lucky Numbers

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

Everybody knows that lucky numbers are positive integers that contain only lucky digits 4 and 7 in their decimal representation. For example, numbers 47, 744, 4 are lucky and 5, 17, 467 are not.

Polo the Penguin have two positive integers l and r ($l < r$), both of them are lucky numbers. Moreover, their lengths (that is, the number of digits in the decimal representation without the leading zeroes) are equal to each other.

Let's assume that n is the number of distinct lucky numbers, each of them cannot be greater than r or less than l, and a_i is the i-th (in increasing order) number of them. Find $a_1 \cdot a_2 + a_2 \cdot a_3 + \ldots + a_{n-1} \cdot a_n$. As the answer can be rather large, print the remainder after dividing it by 1000000007 ($10^9 + 7$).

Input

The first line contains a positive integer l, and the second line contains a positive integer r ($1 \le l < r \le 10^{100000}$). The numbers are given without any leading zeroes.

It is guaranteed that the lengths of the given numbers are equal to each other and that both of them are lucky numbers.

Output

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

Examples

input

```
4
7
```

output

```
28
```

input

```
474
777
```

output

```
2316330
```