Codeforces Round #175 (Div. 2)

A. Slightly Decreasing Permutations

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

Permutation p *is an ordered set of integers* $p_1, p_2, \ldots, p_n$*, consisting of* n *distinct positive integers, each of them doesn't exceed* n*. We'll denote the* i*-th element of permutation* p *as* p_i*. We'll call number* n *the size or the length of permutation* $p_1, p_2, \ldots, p_n$*.*

The decreasing coefficient of permutation $p_1, p_2, ..., p_n$ is the number of such i $(1 \le i < n)$, that $p_i > p_{i+1}$.

You have numbers n and k. Your task is to print the permutation of length n with decreasing coefficient k.

Input

The single line contains two space-separated integers: n, k $(1 \le n \le 10^5, 0 \le k < n)$ — the permutation length and the decreasing coefficient.

Output

In a single line print n space-separated integers: $p_1, p_2, ..., p_n$ — the permutation of length n with decreasing coefficient k.

If there are several permutations that meet this condition, print any of them. It is guaranteed that the permutation with the sought parameters exists.

Sample test(s)

input
5 2
output
1 5 2 4 3

input
3 0
output
1 2 3

input
3 2
output
3 2 1

B. Find Marble

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

Petya and Vasya are playing a game. Petya's got n non-transparent glasses, standing in a row. The glasses' positions are indexed with integers from 1 to n from left to right. Note that the positions are indexed but the glasses are not.

First Petya puts a marble under the glass in position s. Then he performs some (possibly zero) shuffling operations. One shuffling operation means moving the glass from the first position to position p_1, the glass from the second position to position p_2 and so on. That is, a glass goes from position i to position p_i. Consider all glasses are moving simultaneously during one shuffling operation. When the glasses are shuffled, the marble doesn't travel from one glass to another: it moves together with the glass it was initially been put in.

After all shuffling operations Petya shows Vasya that the ball has moved to position t. Vasya's task is to say what minimum number of shuffling operations Petya has performed or determine that Petya has made a mistake and the marble could not have got from position s to position t.

Input

The first line contains three integers: n, s, t $(1 \le n \le 10^5; 1 \le s, t \le n)$ — the number of glasses, the ball's initial and final position. The second line contains n space-separated integers: $p_1, p_2, ..., p_n$ $(1 \le p_i \le n)$ — the shuffling operation parameters. It is guaranteed that all p_i's are distinct.

Note that s can equal t.

Output

If the marble can move from position s to position t, then print on a single line a non-negative integer — the minimum number of shuffling operations, needed to get the marble to position t. If it is impossible, print number -1.

Sample test(s)

input
4 2 1 2 3 4 1
output
3

input
4 3 3 4 1 3 2
output
0

input
4 3 4 1 2 3 4
output
-1

input
3 1 3 2 1 3
output
-1

C. Building Permutation

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

Permutation *p is an ordered set of integers $p_1, p_2, \ldots, p_n$, consisting of n distinct positive integers, each of them doesn't exceed n. We'll denote the i-th element of permutation p as p_i. We'll call number n the size or the length of permutation $p_1, p_2, \ldots, p_n$.*

You have a sequence of integers $a_1, a_2, \ldots, a_n$. In one move, you are allowed to decrease or increase any number by one. Count the minimum number of moves, needed to build a permutation from this sequence.

Input

The first line contains integer n ($1 \le n \le 3 \cdot 10^5$) — the size of the sought permutation. The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$).

Output

Print a single number — the minimum number of moves.

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the cin, cout streams or the `%I64d` specifier.

Sample test(s)

input
2 3 0

output
2

input
3 -1 -1 2

output
6

Note

In the first sample you should decrease the first number by one and then increase the second number by one. The resulting permutation is $(2, 1)$.

In the second sample you need 6 moves to build permutation $(1, 3, 2)$.

D. Permutation Sum

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

Permutation *p is an ordered set of integers $p_1, p_2, \ldots, p_n$, consisting of n distinct positive integers, each of them doesn't exceed n. We'll denote the i-th element of permutation p as p_i. We'll call number n the size or the length of permutation $p_1, p_2, \ldots, p_n$.*

Petya decided to introduce the sum operation on the set of permutations of length n. Let's assume that we are given two permutations of length n: $a_1, a_2, ..., a_n$ and $b_1, b_2, ..., b_n$. Petya calls the sum of permutations a and b such permutation c of length n, where $c_i = ((a_i - 1 + b_i - 1) \bmod n) + 1$ $(1 \le i \le n)$.

Operation $x \bmod y$ means taking the remainder after dividing number x by number y.

Obviously, not for all permutations a and b exists permutation c that is sum of a and b. That's why Petya got sad and asked you to do the following: given n, count the number of such pairs of permutations a and b of length n, that exists permutation c that is sum of a and b. The pair of permutations x, y $(x \ne y)$ and the pair of permutations y, x are considered distinct pairs.

As the answer can be rather large, print the remainder after dividing it by 1000000007 $(10^9 + 7)$.

Input

The single line contains integer n $(1 \le n \le 16)$.

Output

In the single line print a single non-negative integer — the number of such pairs of permutations a and b, that exists permutation c that is sum of a and b, modulo 1000000007 $(10^9 + 7)$.

Sample test(s)

input
3

output
18

input
5

output
1800

E. Positions in Permutations

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

Permutation p *is an ordered set of integers* $p_1, p_2, \ldots, p_n$*, consisting of* n *distinct positive integers, each of them doesn't exceed* n*. We'll denote the* i*-th element of permutation* p *as* p_i*. We'll call number* n *the size or the length of permutation* $p_1, p_2, \ldots, p_n$*.*

We'll call position i ($1 \le i \le n$) in permutation $p_1, p_2, \ldots, p_n$ *good*, if $|p[i] - i| = 1$. Count the number of permutations of size n with exactly k good positions. Print the answer modulo 1000000007 ($10^9 + 7$).

Input

The single line contains two space-separated integers n and k ($1 \le n \le 1000, 0 \le k \le n$).

Output

Print the number of permutations of length n with exactly k good positions modulo 1000000007 ($10^9 + 7$).

Sample test(s)

input
1 0
output
1

input
2 1
output
0

input
3 2
output
4

input
4 1
output
6

input
7 4
output
328

Note

The only permutation of size 1 has 0 good positions.

Permutation $(1, 2)$ has 0 good positions, and permutation $(2, 1)$ has 2 positions.

Permutations of size 3:

1. $(1, 2, 3)$ — 0 positions
2. $(1, \mathbf{3}, \mathbf{2})$ — 2 positions
3. $(\mathbf{2}, \mathbf{1}, 3)$ — 2 positions
4. $(\mathbf{2}, \mathbf{3}, 1)$ — 2 positions
5. $(3, \mathbf{1}, \mathbf{2})$ — 2 positions
6. $(3, 2, 1)$ — 0 positions