Problem 5 – Sorting

You are given a list of numbers containing a permutation of the integers between **1** and **N**, inclusive.

You are allowed to make only **one operation at a time** with the numbers: Take any **K** consecutive elements of the list and **reverse** their order.

You need to **sort the list in ascending order**.

Return the **fewest number of operations** necessary to sort the numbers in ascending order, or **-1** if it's impossible.

Input

The input data should be read from the console.

On the first input line there will be the number **N**.

On the second line there will be **N** numbers separated by a single space (‘ ‘). These numbers represent the initial list which you should sort in ascending order.

On the third line there will be the number **K**.

The input data will always be valid and in the format described. There is no need to check it explicitly.

Output

The output data should be printed on the console.

On the only output line write the fewest number of operations necessary to sort the list of numbers.

Constraints

- **N** and **K** will be integers between 2 and 8, inclusive.
- **K** will always be less than or equal to **N**.
- Allowed working time for your program: 0.20 seconds.
- Allowed memory: 32 MB.

Examples

Example input	Example output	Explanation
3 1 2 3 3	0	The list is already sorted so we don’t need to reorder the numbers.
3 3 2 1 3	1	We need just to reverse the list with one operation.
5 5 4 3 2 1 2	10	4 operations to push 1 to the most left position **1** 5 4 3 2 3 operations to push 2 to the second position 1 **2** 5 4 3 2 operations to push 3 to the third position 1 2 **3** 5 4 1 operations to swap 5 and 4

5 3 2 4 1 5 4	-1	If we can reorder only 4 numbers there is no way to sort this list.
8 7 2 1 6 8 4 3 5 4	7	This permutation of 8 numbers can be sorted with at least 7 operation of reordering 4 numbers at a time.