Not Decreasing

SEASON 2021/2022 – SIXTH ROUND

For the success of task **`increasing`** from this year's first selective contests for the extended junior and girls' national teams, Sashka had to generate the best possible tests for it. For the next test, she decided to use her favourite sequence $a_0, a_1, a_2, \ldots, a_{N-1}$ of N numbers. For every element in the sequence, $1 \leq a_i \leq P$. For every $1 \leq x \leq P$, there is at least one element in the sequence a_i, equal to x. For generating the test, Sashka must execute the code written below. The favourite sequence of Sashka is given in the array `a[]`. The number of elements in the sequence N is given in the parameter `N`, and the number of different values in the array P – in the parameter `P`. The test is contained in the vector b.

```
int cnt[100000 + 1]; // cnt = {0,0,0,…,0}
std::vector < int > getB(int a[], int N, int P)
{
    std::vector < int > b(N, 0);
    for (int i = 0 ; i <= N-1 ; i++)
    {
        cnt[a[i]]++;
        int minCnt = cnt[1];
        for (int j = 2 ; j <= P ; j++)
        {
            minCnt = std::min(minCnt, cnt[j]);
        }
        b[i] = minCnt;
    }
    return b;
}
```

More precisely, the function returns a vector b, in which the i-th value is equal to the minimum number of occurrences of a value $1 \leq x \leq P$ among the first $i + 1$ elements of the sequence. For $N = 7, P = 3$ and $a = \{3,2,1,3,2,2,1\}$, $b = \{0,0,1,1,1,1,2\}$. Sashka generated a wonderful test, but was surprised to discover that she had somehow managed to delete the file containing her favourite sequence. Now, she only has the vector b available. Sashka, outraged by the way she managed to trick herself, wants to recover a by any means. Because of this, she decided to give the problem to the sixth round of CodeIT, hoping at least one of the contestants would find her favorite sequence. To help the contestants in the sixth round, Sashka hints that of all the sequences that correspond to the values in the vector b, her favorite one is that with the shortest possible ***longest non-decreasing subsequence***. The *longest non-decreasing subsequence* of a sequence is obtained by deleting some elements (potentially 0) from the sequence and preserving the order of the remaining elements, so that the number of remaining elements is as large as possible and they form a non-decreasing sequence. A sequence is non-decreasing if there is no pair of elements in it for which the left element is greater than the right element. One more condition

Not Decreasing

SEASON 2021/2022 – SIXTH ROUND

from Sashka is that for every value $1 \leq x \leq P$, the sequence must contain at least one element equal to x.

After too much explaining, you are left with the honour of writing a program `notdecreasing.cpp` that finds a row a that corresponds to the values in b and has the shortest possible ***longest non-decreasing subsequence***. Sashka would accept any sequence that satisfies the conditions.

Input

The first line of `notdecreasing.in` contains the integers N and P. The second line of the file contains N integers, $b_0, b_1, b_2, \ldots, b_{N-1}$ respectively.

Output

On the first line of `notdecreasing.out` print N integers, $a_0, a_1, a_2, \ldots, a_{N-1}$ respectively. If there is more than one sequence, satisfying the requirements, you may print any of them. If there aren't any sequences, satisfying the requirements, you should print -1.

Constraints

$1 \leq P \leq N \leq 10^5$

$0 \leq b_i \leq 10^9$

Time Limit: 0.5 sec.
Memory Limit: 256 MB.

Sample tests

Input (notdecreasing.in)	Output (notdecreasing.out)
4 2 0 1 1 2	2 1 2 1
4 2 0 1 1 0	-1
7 3 0 0 1 1 1 1 2	3 2 1 3 2 2 1

The given sample outputs are only for illustration and there can be more sequences, satisfying the requirements.