I LOVE Kd-TREES III

The "I-Love-Kd-trees" annual con is receiving too many applicants so they decided to complicate a bit the task used to select participants. (They realized some people were using other data structures to solve their problems, so they designed this problem, almost only solvable with Kd-trees).

You are given a list of **N** numbers and **Q** queries.

Each query can be of two types.

Type-0 queries (marked with 0 in the input), consist of three integers: **i, l** and **k** ; let **d** be the **k-th** smallest element until the index **i** (i.e. if the first i +1 elements were sorted in non-descending way, **d** would be the element at index k - 1). Then, the answer to each query is the index of the **l-th** occurrence of **d** in the array. If there's no such index, the answer is -1.

Type-1 queries (marked with 1 in the input) are contiguous-swap update-queries, and consists of a single integer **i**. When a type-1 query is executed the elements at index i and i +1 in the list must be swapped.

You have to consider that all indexes are counted starting with 0.

Input

Input consists of one test case.

The first line contains two integers, **N** ($1 \leq N \leq 10^6$) and **Q** ($1 \leq Q \leq 10^5$).

The next line contains N possible distinct integers $\mathbf{a_i}$ ($-10^9 \leq ai \leq 10^9$).

Then **Q** lines follow. Each of them starts with an integer which can be 0 or 1, denoting the type of the query. If it's 0, then three integers **i**, **l** and **k** follow ($0 \leq i < N$, $1 \leq k \leq i+1$, $1 \leq l \leq N$).

If it's 1, then an integer **i** follows, meaning that you have to swap the elements at indexes i and i+1 ($0 \leq i \leq N-1$).

Output

For each query of type-0 (in the same order as the input) output a single line with the answer to that query.

Example

Input:

```
10 6
2 3 1 1 2 7 9 1 2 6
0 2 3 2
1 1
1 2
0 2 3 2
```

1 0
0 0 2 1
Output:
8
7
2