Problem Clay. Order Statistics

Input file:	`input.txt` or standard input
Output file:	`output.txt` or standard output
Time limit:	4 seconds
Memory limit:	1024 megabytes

You are given an array $a_1, a_2, \ldots, a_n$, consisting of integers, as well as integers k and m. The following operation is performed m times on the array:

- Select $i_1, i_2, \ldots, i_k$ — the positions of the k largest elements in the array a. If two elements are equal, the element that appears earlier in the array is considered larger.
- Decrease $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ by 1.

For x from 1 to n, let $F_{m,k}(x)$ denote the value of the x-th order statistic in the array obtained from a after applying the operation m times with the given parameter k. For x from 1 to n, the x-th order statistic of the array $a_1, a_2, \ldots, a_n$ is the element that would be in position x if the array a were sorted in non-decreasing order.

For all l, r such that $1 \le l \le r \le n$, let $S_{m,k}(l, r)$ denote the sum of $F_{m,k}(x)$ for all integers x from l to r. More formally:

$$S_{m,k}(l,r) = \sum_{x=l}^{r} F_{m,k}(x)$$

You are given integers m_0 and k_0. You must compute the values of $F_{m_0,k_0}(x)$ for all x from 1 to n.

After that, you must process q queries. The j-th query $(1 \le j \le q)$ can be one of three types:

1. Compute the value of $F_{m_j,k_j}(x_j)$.
2. Change the value of a_{p_j} to v_j.
3. Compute the value of $S_{m_j,k_j}(l_j, r_j)$.

All calculations of the functions F and S are performed independently each time and do not change the array. All changes to the array in queries of the second type are preserved for subsequent queries.

Input

The first line contains four integers n, m_0, k_0, and q ($1 \le n \le 200\,000$, $0 \le m_0 \le 10^9$, $1 \le k_0 \le n$, $0 \le q \le 200\,000$) — the length of the array a; the number of operations; the number of largest elements decreased in each operation, and the number of queries.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$, $1 \le i \le n$) — the elements of the array a.

The next q lines contain the queries. In the j-th query, the first number is t_j ($1 \le t_j \le 3$) — the type of the j-th query.

- If $t_j = 1$, then the next line contains three integers m_j, k_j, and x_j ($0 \le m_j \le 10^9$, $1 \le k_j, x_j \le n$) — the parameters of the first type query.
- If $t_j = 2$, then the next line contains two integers p_j and v_j ($1 \le p_j \le n$, $-10^9 \le v_j \le 10^9$) — the parameters of the second type query.
- If $t_j = 3$, then the next line contains four integers m_j, k_j, l_j, and r_j ($0 \le m_j \le 10^9$, $1 \le k_j, l_j, r_j \le n$, $l_j \le r_j$) — the parameters of the third type query.

Output

In the first line, output n integers $F_{m_0,k_0}(1), F_{m_0,k_0}(2), \ldots, F_{m_0,k_0}(n)$.

Then, for each first type query, output the value $F_{m_j,k_j}(x_j)$ on a separate line, and for each third type query, output the value $S_{m_j,k_j}(l_j, r_j)$ — the answer to the j-th query.

Example

input	output

`8 3 2 16`
`3 1 2 -1 0 2 -1 4`
`3 3 2 2 6`
`1 3 2 4`
`3 4 5 3 5`
`1 4 5 6`
`2 5 -1`
`2 6 3`
`1 3 2 1`
`1 3 2 3`
`1 3 2 4`
`1 3 2 8`
`1 0 5 6`
`2 1 5`
`3 1 3 7 8`
`3 2 3 5 8`
`3 3 3 4 7`
`3 4 3 4 7`
	`-1 -1 0 1 1 1 1 2`
	`2`
	`1`
	`-4`
	`-1`
	`-1`
	`-1`
	`1`
	`2`
	`3`
	`7`
	`8`
	`4`
	`2`

Note

In the example, $n = 8$, $m_0 = 3$, $k_0 = 2$, $q = 16$. Initially, the array a is $[3, 1, 2, -1, 0, 2, -1, 4]$. Let's see how the array will change if we apply the operation m_0 times with parameter k_0:

1. The array is $[3, 1, 2, -1, 0, 2, -1, 4]$. The two largest elements are in positions 1 and 8. They are decreased by 1, after which the array becomes $[2, 1, 2, -1, 0, 2, -1, 3]$.
2. The array is $[2, 1, 2, -1, 0, 2, -1, 3]$. The two largest elements are in positions 1 and 8. They are decreased by 1, after which the array becomes $[1, 1, 2, -1, 0, 2, -1, 2]$.
3. The array is $[1, 1, 2, -1, 0, 2, -1, 2]$. The two largest elements are in positions 3 and 6. They are decreased by 1, after which the array becomes $[1, 1, 1, -1, 0, 1, -1, 2]$.

We find that after applying the operation 3 times with parameter 2 to the array a, it becomes $[1, 1, 1, -1, 0, 1, -1, 2]$. If this array is sorted, it results in the array $[-1, -1, 0, 1, 1, 1, 1, 2]$. Thus, the order statistics are $F_{3,2}(1) = -1$, $F_{3,2}(2) = -1$, $F_{3,2}(3) = 0$, $F_{3,2}(4) = 1$, $F_{3,2}(5) = 1$, $F_{3,2}(6) = 1$, $F_{3,2}(7) = 1$, $F_{3,2}(8) = 2$.

In the example, we need to process 16 queries; let's analyze the first 10 of them in detail:

1. The first query is of type $t_1 = 3$, with parameters $m_1 = 3$, $k_1 = 2$, $l_1 = 2$, $r_1 = 6$ and requires computing the value of $S_{3,2}(2, 6)$. We have already computed the values of $F_{3,2}(x)$ for x from 1 to 8, from which we find the answer to the query:
$$S_{3,2}(2, 6) = F_{3,2}(2) + F_{3,2}(3) + F_{3,2}(4) + F_{3,2}(5) + F_{3,2}(6) = (-1) + 0 + 1 + 1 + 1 = 2$$
2. The second query is of type $t_2 = 1$, with parameters $m_2 = 3$, $k_2 = 2$, $x_2 = 4$ and requires computing the value of $F_{3,2}(4)$. We have already computed it, and it equals 1.

3. The third query is of type $t_3 = 3$, with parameters $m_3 = 4$, $k_3 = 5$, $l_3 = 3$, $r_3 = 5$ and requires computing the value of $S_{4,5}(3, 5)$, that is, to calculate the sum of the order statistics from the third to the fifth in the array obtained from a after applying the operation $m_3 = 4$ times with parameter $k_3 = 5$. At the time of the third query, the array a is $[3, 1, 2, -1, 0, 2, -1, 4]$. The five largest elements are in positions $1, 2, 3, 6, 8$. Decreasing them by 1, we get the array $[2, 0, 1, -1, 0, 1, -1, 3]$. Applying the operation three more times, we get the array $[-1, -2, -2, -2, -1, -1, -1, 0]$. After sorting, it becomes $[-2, -2, -2, -1, -1, -1, -1, 0]$. Thus, the answer to the query is

$$S_{4,5}(3, 5) = F_{4,5}(3) + F_{4,5}(4) + F_{4,5}(5) = (-2) + (-1) + (-1) = -4$$

4. The fourth query is of type $t_4 = 1$, with parameters $m_4 = 4$, $k_4 = 5$, $x_4 = 6$. After applying four operations with parameter 5 and sorting the array a, it will be $[-2, -2, -2, -1, -1, -1, -1, 0]$, so the sixth order statistic is -1.

5. The fifth query is of type $t_5 = 2$, with parameters $p_5 = 5$ and $v_5 = -1$. It changes the value of a_5 to -1, after which the array a becomes $[3, 1, 2, -1, -1, 2, -1, 4]$.

6. The sixth query is of type $t_6 = 2$, with parameters $p_6 = 6$ and $v_6 = 3$. It changes the value of a_6 to 3, after which the array a becomes $[3, 1, 2, -1, -1, 3, -1, 4]$.

7. The seventh query requires finding the value of $F_{3,2}(1)$. At the time of the seventh query, the array a is $[3, 1, 2, -1, -1, 3, -1, 4]$. After applying 3 times the operation with parameter 2, it will be $[1, 1, 1, -1, -1, 2, -1, 2]$. The first order statistic of this array is -1.

8. The eighth, ninth, and tenth queries require finding the values of $F_{3,2}(3)$, $F_{3,2}(4)$ and $F_{3,2}(8)$, that is, the third, fourth, and eighth order statistics in the array $[1, 1, 1, -1, -1, 2, -1, 2]$. They are equal to -1, 1, and 2, respectively.

Scoring

The tests for this problem consist of eleven groups. Points for each group are given only if all tests of the group and all tests of the required groups are passed. Please note that passing the example tests is not required for some groups. **Offline-evaluation** means that the results of testing your solution on this group will only be available after the end of the competition.

If there are constraints on m or k in a subtask, they apply to both m_0 and k_0, as well as to the parameters of all first and third type queries.

The table with the groups is on the next page.

Group	Points	Additional Constraints				Required Groups	Comment
		n	m	k	q		
0	0	–	–	–	–	–	Examples.
1	4	$n \le 1000$	$m \le 1000$	–	$q = 0$	–	–
2	5	–	–	$k = 1$	$q = 0$	–	–
3	6	–	–	$k = 1$	$q \le 100\,000$	2	$t_j = 1$ for all queries
4	7	–	–	$k = 1$	$q \le 100\,000$	2, 3	$t_j \ne 3$ for all queries
5	11	–	–	$k = 2$	$q = 0$	–	–
6	9	–	$m \le 10^6$	–	$q = 0$	1	–
7	10	$n \le 1000$	–	–	$q = 0$	1	–
8	7	–	–	–	$q = 0$	1, 2, 5 – 7	–
9	11	–	–	–	$q \le 100\,000$	1 – 3, 5 – 8	$t_j = 1$ for all queries
10	13	–	–	–	$q \le 100\,000$	1 – 3, 5 – 9	$t_j \ne 2$ for all queries
11	9	–	–	–	$q \le 100\,000$	0 – 10	–
12	8	–	–	–	–	0 – 11	**Offline-evaluation.**