Задача A. Fence Painting

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	1 second
Ограничение по памяти:	512 megabytes

Tom and Huck are painting a fence. The fence consists of $a + b$ segments, with the first a segments being wooden and the remaining b being metal.

Tom and Huck have been promised a reward for painting the fence. Since different amounts of effort are required for wooden and metal segments, painting a wooden segment is valued at x coins, and a metal segment at y coins.

Tom starts painting the fence from the first (far left) segment and paints one segment after another in succession. Similarly, Huck starts painting from the right side and moves towards Tom. Naturally, the boys want the reward for painting the fence to be divided fairly: the amounts that Tom and Huck receive for the fence should differ by the smallest possible value.

Help them determine their meeting point k, so that the reward for painting is as evenly distributed as possible. The meeting point is considered to be the number of the last segment painted by Tom. In other words, if the meeting point is k, Tom has painted the k left segments, and Huck has painted the remaining $a + b - k$ right segments.

Формат входных данных

The first line of input contains two integers a and b — the number of wooden and metal fence segments, respectively ($1 \leqslant a, b \leqslant 10^9$).

The second line of input contains two integers x and y — the number of coins paid for painting one wooden or metal segment, respectively ($1 \leqslant x, y \leqslant 10^9$).

Формат выходных данных

Output a single integer k — the meeting point where Tom and Huck receive the most closely valued rewards. If there are several optimal answers, output any of them.

Система оценки

Points for each subtask are awarded only if all tests of that subtask and the required subtasks are successfully passed.

Subtask	**Points**	**Constraints**	**Required subtasks**
0	–	examples	
1	16	$a, b \leqslant 1000$	0
2	19	$x = y$	
3	21	$a = b$	
4	24	$a, b \leqslant 10^6$	0, 1
5	20	none	0 – 4

Примеры

standard input	standard output
1 1 1 1	1
5 10 2 1	5

Задача B. Work, Sleep, Repeat

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	1 second
Ограничение по памяти:	256 megabytes

Very talented programmers work at the VK company. Since they constantly have to solve interesting problems, work goes by quickly and with pleasure. However, this does not mean that one can completely forgo rest and sleep from it.

Programmer Lev who (almost) holds the position of a team lead decided to take a vacation. But since his presence at the workplace and availability can be critical for the team he will split this vacation into several parts: namely, he will live in a cyclical mode of "work for x days, then take a vacation for y days" for the foreseeable future.

Of course, he has planned meetings with the team, release days for new versions, and other events that cannot be missed. There are a total of n such events and the i-th of them is scheduled for day d_i.

Assuming that today is day number 1, and all events are scheduled for no earlier than day number $x + y + 1$, determine if there exists a day $D \leqslant x + y$ such that if Lev switches to the new mode on day D, he will not miss any scheduled events. In other words, it is required to ensure that all days d_i fall on the work phase in the corresponding cycle.

Формат входных данных

The first line of input contains three space-separated integers n, x and y — the number of scheduled events, as well as the length of the work phase and the length of the vacation phase planned by Lev ($1 \leqslant n \leqslant 2 \cdot 10^5$; $1 \leqslant x, y \leqslant 10^9$).

The second line lists n space-separated integers d_i — the days on which events are scheduled ($x + y < d_i \leqslant 10^9$). It is **not** guaranteed that all d_i are different.

Формат выходных данных

Output a single integer between 1 and $x + y$ — the day number on which Lev should switch to the new mode. If there are several possible answers, output any of them. If there is no answer, output `-1`.

Система оценки

Points for each subtask are awarded only if all tests of that subtask and the required subtasks are successfully passed.

Subtask	Points	Constraints	Required subtasks
0	–	examples	
1	15	$x = 1$	
2	19	$n, x, y \leqslant 1000$	0
3	18	$y = 1$	
4	21	$x \leqslant y$	0, 2
5	27	none	0 – 4

Примеры

standard input	standard output
3 3 1 5 8 11	3
4 3 1 5 8 11 14	-1
5 4 3 10 16 17 18 99	1

Замечание

In the first example if the cycle starts on the third day Lev will work on days 3–5, then 7, 8, 9, and finally, 11–13.

Задача C. Palindromization

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 megabytes

Palindromes are sequences that read the same both left-to-right and right-to-left. Such sequences play a significant role in string algorithms. Programmer Vlad believes that working with palindromes is easier than with any other strings or arrays, so whenever Vlad sees an array, he strives to turn it into a palindrome.

This time, Vlad received an array a of n integers. In one operation, Vlad can choose any segment of this array and add the same number x from 1 to k inclusive to all elements on this segment.

Despite being accustomed to working with palindromes, this time Vlad encountered problems with turning a into a palindrome with the minimum number of actions. Help him quickly cope with this task.

Формат входных данных

The first line of input contains two integers n and k — the size of the original array and the limit on the added values ($1 \leqslant n \leqslant 10^6$; $1 \leqslant k \leqslant 3$).

The second line of input contains n space-separated integers a_i — the elements of the array ($1 \leqslant a_i \leqslant 10^9$).

Формат выходных данных

Output a single number — the minimum number of operations required to turn the array a into a palindrome.

Система оценки

Points for each subtask are awarded only if all tests of that subtask and the required subtasks are successfully passed.

Subtask	**Points**	**Constraints**	**Required subtasks**
0	–	examples	
1	10	$n, a_i \leqslant 10$	0
2	11	$a_i \leqslant 2$, $k = 1$	
3	12	$a_i \leqslant a_{i+1}$ for all $i < n$	
4	20	$k = 1$	2
5	22	$k \leqslant 2$	4
6	25	$k \leqslant 3$, $n \leqslant 1000$	0, 1

Notice that there is no group with $k = 3$ and no additional constraints!

Примеры

standard input	standard output
5 1 1 2 3 4 5	4
6 2 3 6 4 1 2 5	4
8 3 1 4 3 1 2 1 1 2	3

Замечание

In the first example, you need to add 1 to the segment $[1, 3]$, add 1 to the segment $[1, 2]$, and then add 1 twice to the first element. Thus, after four operations, the array equals $[5, 4, 4, 4, 5]$.

In the second example, you need to add 2 to the segment $[1, 1]$, add 2 to the segment $[4, 5]$ and add 1 to the segment $[4, 5]$, after which add 1 to the segment $[5, 5]$. Thus, after four operations, the array equals $[5, 6, 4, 4, 6, 5]$.

In the third example, you need to add 1 to the segment $[1, 4]$, add 3 to the segment $[6, 7]$ and add 1 to the segment $[7, 7]$. Thus, after three operations, the array equals $[2, 5, 4, 2, 2, 4, 5, 2]$.

Задача D. Hackathon

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	2 seconds
Ограничение по памяти:	512 megabytes

VK often holds hackathons where programmers solve various problems. VK is now preparing a new hackathon in which anyone can try their hand at developing a unique and useful application or service for one of several tracks including Fintech, Culture, Media, and others.

As with any well-organized hackathon, to allow participants to fully dedicate themselves to solving the task at hand they are provided with a carefully organized venue. Pleasant bonuses often include food and drinks, which can be enjoyed over the several days that participants spend at the venue.

This time VK is organizing a new hackathon, and m participants have gathered at the venue. For these m participants the organizers have ordered n pizzas which are arranged in a row and numbered from 1 to n. It is known that:

- each pizza consists of an infinite number of slices;
- the first slice of the i-th pizza has a dough thickness of a_i, and each subsequent one is 1 thicker than the previous (i.e., $a_i + 1$, $a_i + 2$, and so on);
- the slices with the least dough thickness seem most attractive to participants;
- any participant can approach the row of pizzas at any point, but afterwards can only move left (towards the pizzas with smaller numbers) to avoid collisions with other participants.

Periodically, participants get up from their places and approach the row of pizzas to take a slice. Participant number i approaches the pizza numbered s_i at time t_i, after which they immediately find the slice of pizza with the least dough thickness among pizzas from 1 to s_i and start moving towards it. It takes each participant the same amount of time to move between two adjacent pizzas — exactly v seconds. Several participants can be near the same pizza at the same time.

As soon as a participant reaches the pizza they have chosen, they take the slice and return to their seat. If several participants have chosen the same slice and reach it simultaneously, the participant with the lower number takes the slice. If the slice a participant was counting on is taken by someone else, that participant immediately chooses a new target using the same algorithm: they look at all the pizzas from the first to their current position, select the minimum among them, and start moving towards it.

Formally, this process can be described as follows. At every whole number of seconds t:

1. all participants who were already in the row at time $t - 1$ move by $\frac{1}{v}$ to the left;
2. all participants with $t_i = t$ approach the pizza number s_i;
3. time "stops";
4. all participants standing at **whole** positions p such that $a_p = \min(a_1, a_2, \ldots, a_p)$ are identified;
5. for each such p, the participant with the minimum number standing at position p takes the slice of pizza with thickness a_p and leaves; a_p is increased by 1;
6. after some slices of pizza are taken, it may happen that the minimum thickness of the slice that some participants can still get has changed, in which case the process repeats from step 4;
7. otherwise, time "starts again" and the process moves to the next second.

Based on the information about participants' approaches to the pizzas, determine for each participant the thickness of the pizza slice they received and how much time they spent in the row.

Формат входных данных

The first line of input contains three space-separated integers n, m, and v — the number of pizzas, the number of participants and the time it takes each participant to move between adjacent pizzas ($1 \leqslant n, m \leqslant 10^5$; $1 \leqslant v \leqslant 10^9$).

The second line lists n space-separated integers a_i — the thickness of the first slice of each pizza ($0 \leqslant a_i \leqslant 10^9$).

In the i-th of the following m lines, two integers s_i and t_i are given, separated by a space — the starting position and the time of the participant's approach to the row of pizzas ($1 \leqslant s_i \leqslant n$; $0 \leqslant t_i \leqslant 10^9$).

Формат выходных данных

Output m lines, in the i-th of which print a pair of space-separated integers: the dough thickness of the pizza slice received by the i-th participant and the time it took the participant to reach it.

Система оценки

Points for each subtask are awarded only if all tests of that subtask and the required subtasks are successfully passed.

Subtask	Points	Constraints	Required subtasks
0	–	examples	
1	8	$\max(t_i) \leqslant 10^4$; $n, m, v \leqslant 100$	0
2	10	all s_i are equal and all t_i are equal	
3	13	$\lvert a_i - a_j \rvert \geqslant m$ for any $i \neq j$	
4	19	$n, m \leqslant 1000$	0, 1
5	16	$m \leqslant 1000$	0, 1, 4
6	34	none	1 – 5

Примеры

standard input	standard output
5 3 1 1 2 3 4 5 5 1 3 2 2 4	2 3 1 2 2 1
7 7 3 1 5 3 2 5 4 1 5 4 2 3 4 2 1 6 7 8 3 2 2 4	2 3 1 3 5 6 2 0 1 0 4 6 3 3

Замечание

In the first example, events will occur in the following order:

1. The first participant appears at position 5 at time 1; their target is pizza number 1.
2. By the time 2, the first participant is at position 4, and the second participant appears at position 3. Both will move towards pizza number 1.

3. In the third second, no changes occur, and the participants continue to move left.

4. At time 4: the second participant is near the first pizza, the first and third participants are together near the second. The second participant takes a slice of pizza with a thickness of 1, a_1 becomes equal to 2.

5. The first and third participants immediately see that they will no longer get a slice of pizza with a thickness of 1, but they can take one with a thickness of 2 at their current position. The first, as the participant with the lower number, takes $a_2 = 2$, after which a_2 becomes equal to 3.

6. The remaining third participant can only move one more to the left and take $a_1 = 2$.

Задача E. Tree Trisection

Имя входного файла:	`standard input`
Имя выходного файла:	`standard output`
Ограничение по времени:	4 seconds
Ограничение по памяти:	256 megabytes

On the territory of Innopolis University there are many trees that require maintenance. In particular, Innopolis has weighted full binary trees that sometimes need to be pruned to prevent them from growing too quickly.

Consider a full binary tree with $n = 2^k$ leaves. Let's assign the root number 1, its children numbers 2 and 3 (from left to right) and so on, numbering each layer of vertices with consecutive integers. The leaves of the tree will have numbers from n to $2n - 1$. The weight of a vertex number i is equal to w_i. The weight of a set of vertices is defined as the sum of the weights of the vertices in that set.

Your task is to determine how to prune the trees efficiently. A *pruning* request is specified by the leaf numbers l and r, and is carried out as follows:

1. First, a set of vertices V is selected — this is the smallest connected set of vertices that contains **all** the leaves from the l-th to the r-th. Let's denote the vertex of the set V that is closest to the root of the tree as $\texttt{root}(V)$.

2. Then, it is required to select in the set V two different vertices x and y such that neither of them is a descendant of the other.

3. Once the vertices x and y are chosen, the edges leading from $\texttt{root}(V)$, x, and y "upwards" (towards the root) are cut. As a result, V is separated from the entire tree and breaks down into three subsets, denoted as $V_{\texttt{root}}$, V_x and V_y, respectively.

4. After that, all three of these subsets are planted as separate sub-trees to enhance the landscaping of the territory.

The management has several options for how to prune the given tree. Your goal is to determine for each of their requests (l, r) which vertices x and y should be chosen to minimize the maximum weight among $V_{\texttt{root}}$, V_x, and V_y.

Формат входных данных

The first line contains two space-separated integers n and m — the number of leaves in the tree and the number of management's requests ($2 \leqslant n \leqslant 2^{16}$; $1 \leqslant m \leqslant 10^5$; $n = 2^k$ for some integer k).

The next line lists $2n - 1$ integers w_i separated by a space — the weights of the tree vertices ($1 \leqslant w_i \leqslant 10^9$).

In the i-th of the following m lines the i-th management's request is given as two integers l_i and r_i — the number of the first and last leaves involved in the pruning of the tree ($n \leqslant l_i, r_i \leqslant 2n-1$; $r_i \geqslant l_i + 1$).

Формат выходных данных

For each request, output in a separate line two space-separated integers x and y — the numbers of additional vertices where the tree should be cut to minimize the maximum weight of the three cut parts.

If there are several options for answers that minimize the maximum weight of the cut part, output any of them.

Система оценки

By default, points for each subtask are awarded only if all tests of that subtask and the required subtasks are successfully passed. Points for a subtask with per-test scoring are awarded for each passed test independently (2 points for each of the sixteen tests).

Subtask	Points	Constraints	Required subtasks	Scoring
0	–	examples		–
1	6	$n \leqslant 32$, $m \leqslant 20$	0	full group
2	10	$n \leqslant 512$, $m \leqslant 400$	0, 1	full group
3	13	$r_i = l_i + 1$ for all i		full group
4	23	$w_i = 1$ for all i		full group
5	16	$n \leqslant 1024$, $m \leqslant 2000$	0 – 2	full group
6	16 × 2	none	0 – 5	test-wise

Примеры

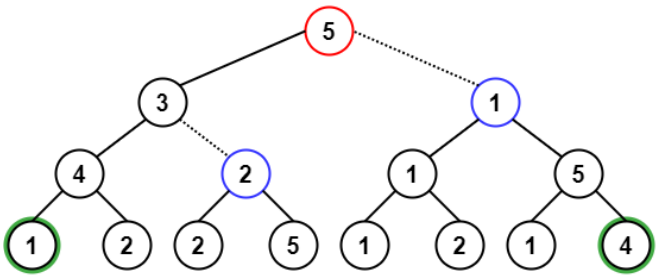
standard input	standard output
4 4 1 1 1 1 1 1 1 4 5 5 6 6 7 4 7	4 5 5 3 6 7 5 3
8 7 5 3 1 4 2 1 5 1 2 2 5 1 2 1 4 8 15 8 11 10 13 8 12 11 12 9 14 13 15	5 3 4 11 5 3 4 5 5 3 5 3 6 15

Замечание

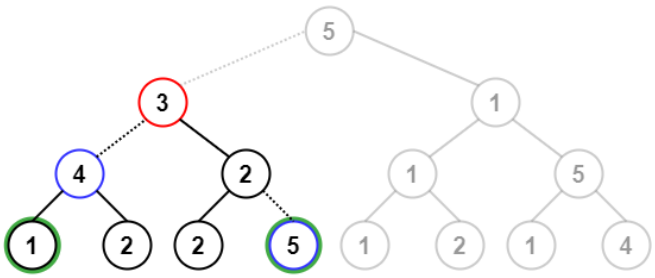
Below is an illustration for the first four requests in the second example from the problem statement.

The weight of each vertex is marked. Vertices that are not included in the set V are highlighted in gray. The selected leaves l and r are circled in green. The vertex $\texttt{root}(V)$ is circled in red, and the vertices x and y, which need to be chosen as the answer, are circled in blue. The edges that are cut during the corresponding *pruning* are highlighted with a dotted line.

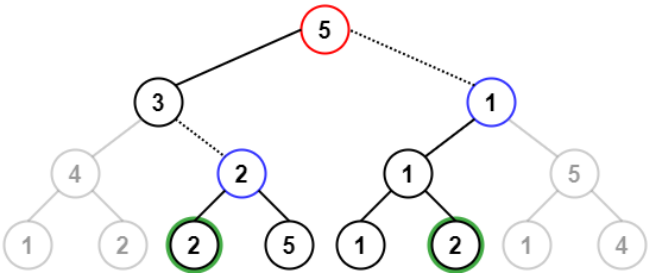
Request 1



Request 2



Request 3



Request 4