Problem A. Equalize the Array

OS Linux

Given an array of integers, determine the minimum number of elements to delete to leave only elements of equal value.

Example

$arr = [1, 2, 2, 3]$

Delete the 2 elements 1 and 3 leaving $arr = [2, 2]$. If both twos plus either the 1 or the 3 are deleted, it takes 3 deletions to leave either $[3]$ or $[1]$. The minimum number of deletions is 2.

Function Description

Complete the *equalizeArray* function in the editor below.

equalizeArray has the following parameter(s):

- *int arr[n]:* an array of integers

Returns

- *int:* the minimum number of deletions required

Input Format

The first line contains an integer n, the number of elements in arr.
The next line contains n space-separated integers $arr[i]$.

Constraints

- $1 \leq n \leq 100$
- $1 \leq arr[i] \leq 100$

Input	Output
STDIN ----- 5 3 3 2 1 3 Function -------- arr[] size n = 5 arr = [3, 3, 2, 1, 3]	2

Explanation

Delete $arr[2] = 2$ and $arr[3] = 1$ to leave $arr' = [3, 3, 3]$. This is minimal. The only other options are to delete 4 elements to get an array of either $[1]$ or $[2]$.

Problem B. Lecture Sleep

Time Limit 1000 ms
Mem Limit 262144 kB

Your friend Mishka and you attend a calculus lecture. Lecture lasts n minutes. Lecturer tells a_i theorems during the i-th minute.

Mishka is really interested in calculus, though it is so hard to stay awake for all the time of lecture. You are given an array t of Mishka's behavior. If Mishka is asleep during the i-th minute of the lecture then t_i will be equal to 0, otherwise it will be equal to 1. When Mishka is awake he writes down all the theorems he is being told — a_i during the i-th minute. Otherwise he writes nothing.

You know some secret technique to keep Mishka awake for k minutes straight. However you can use it **only once**. You can start using it at the beginning of any minute between 1 and $n - k + 1$. If you use it on some minute i then Mishka will be awake during minutes j such that $j \in [i, i + k - 1]$ and will write down all the theorems lecturer tells.

You task is to calculate the maximum number of theorems Mishka will be able to write down if you use your technique **only once** to wake him up.

Input

The first line of the input contains two integer numbers n and k ($1 \le k \le n \le 10^5$) — the duration of the lecture in minutes and the number of minutes you can keep Mishka awake.

The second line of the input contains n integer numbers $a_1, a_2, \dots a_n$ ($1 \le a_i \le 10^4$) — the number of theorems lecturer tells during the i-th minute.

The third line of the input contains n integer numbers $t_1, t_2, \dots t_n$ ($0 \le t_i \le 1$) — type of Mishka's behavior at the i-th minute of the lecture.

Output

Print only one integer — the maximum number of theorems Mishka will be able to write

down if you use your technique **only once** to wake him up.

Examples

Input	Output
6 3 1 3 5 2 5 4 1 1 0 1 0 0	16

Note

In the sample case the better way is to use the secret technique at the beginning of the third minute. Then the number of theorems Mishka will be able to write down will be equal to 16.

Problem C. Greg and Array

Time Limit	1500 ms
Mem Limit	262144 kB
Input File	stdin
Output File	stdout

Greg has an array $a = a_1, a_2, \ldots, a_n$ and m operations. Each operation looks as: l_i, r_i, d_i, $(1 \le l_i \le r_i \le n)$. To apply operation i to the array means to increase all array elements with numbers $l_i, l_i + 1, \ldots, r_i$ by value d_i.

Greg wrote down k queries on a piece of paper. Each query has the following form: x_i, y_i, $(1 \le x_i \le y_i \le m)$. That means that one should apply operations with numbers $x_i, x_i + 1, \ldots, y_i$ to the array.

Now Greg is wondering, what the array a will be after all the queries are executed. Help Greg.

Input

The first line contains integers n, m, k $(1 \le n, m, k \le 10^5)$. The second line contains n integers: $a_1, a_2, \ldots, a_n$ $(0 \le a_i \le 10^5)$ — the initial array.

Next m lines contain operations, the operation number i is written as three integers: l_i, r_i, d_i, $(1 \le l_i \le r_i \le n)$, $(0 \le d_i \le 10^5)$.

Next k lines contain the queries, the query number i is written as two integers: x_i, y_i, $(1 \le x_i \le y_i \le m)$.

The numbers in the lines are separated by single spaces.

Output

On a single line print n integers $a_1, a_2, \ldots, a_n$ — the array after executing all the queries. Separate the printed numbers by spaces.

Please, do not use the `%lld` specifier to read or write 64-bit integers in *C++*. It is preferred to use the `cin`, `cout` streams of the `%I64d` specifier.

Examples

Input	Output
3 3 3 1 2 3 1 2 1 1 3 2 2 3 4 1 2 1 3 2 3	9 18 17

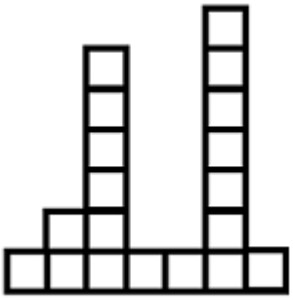
Input	Output
1 1 1 1 1 1 1 1 1	2

Input	Output
4 3 6 1 2 3 4 1 2 1 2 3 2 3 4 4 1 2 1 3 2 3 1 2 1 3 2 3	5 18 31 20

Problem D. Fence

Time Limit	1000 ms
Mem Limit	262144 kB
Input File	stdin
Output File	stdout

There is a fence in front of Polycarpus's home. The fence consists of n planks of the same width which go one after another from left to right. The height of the i-th plank is h_i meters, distinct planks can have distinct heights.



Fence for $n = 7$ and $h = [1, 2, 6, 1, 1, 7, 1]$

Polycarpus has bought a posh piano and is thinking about how to get it into the house. In order to carry out his plan, he needs to take exactly k consecutive planks from the fence. Higher planks are harder to tear off the fence, so Polycarpus wants to find such k consecutive planks that the sum of their heights is minimal possible.

Write the program that finds the indexes of k consecutive planks with minimal total height. Pay attention, the fence is not around Polycarpus's home, it is in front of home (in other words, the fence isn't cyclic).

Input

The first line of the input contains integers n and k ($1 \le n \le 1.5 \cdot 10^5$, $1 \le k \le n$) — the number of planks in the fence and the width of the hole for the piano. The second line contains the sequence of integers $h_1, h_2, \ldots, h_n$ ($1 \le h_i \le 100$), where h_i is the height of the i-th plank of the fence.

Output

Print such integer j that the sum of the heights of planks $j, j+1, \ldots, j+k-1$ is the minimum possible. If there are multiple such j's, print any of them.

Examples

Input	Output
7 3 1 2 6 1 1 7 1	3

Note

In the sample, your task is to find three consecutive planks with the minimum sum of heights. In the given case three planks with indexes 3, 4 and 5 have the required attribute, their total height is 8.

Problem E. Karen and Coffee

Time Limit 2500 ms
Mem Limit 524288 kB

To stay woke and attentive during classes, Karen needs some coffee!

Karen, a coffee aficionado, wants to know the optimal temperature for brewing the perfect cup of coffee. Indeed, she has spent some time reading several recipe books, including the universally acclaimed "The Art of the Covfefe".

She knows n coffee recipes. The i-th recipe suggests that coffee should be brewed between l_i and r_i degrees, inclusive, to achieve the optimal taste.

Karen thinks that a temperature is *admissible* if at least k recipes recommend it.

Karen has a rather fickle mind, and so she asks q questions. In each question, given that she only wants to prepare coffee with a temperature between a and b, inclusive, can you tell her how many admissible integer temperatures fall within the range?

Input

The first line of input contains three integers, n, k ($1 \le k \le n \le 200000$), and q ($1 \le q \le 200000$), the number of recipes, the minimum number of recipes a certain temperature must be recommended by to be admissible, and the number of questions Karen has, respectively.

The next n lines describe the recipes. Specifically, the i-th line among these contains two integers l_i and r_i ($1 \le l_i \le r_i \le 200000$), describing that the i-th recipe suggests that the coffee be brewed between l_i and r_i degrees, inclusive.

The next q lines describe the questions. Each of these lines contains a and b, ($1 \le a \le b \le 200000$), describing that she wants to know the number of admissible integer temperatures between a and b degrees, inclusive.

Output

For each question, output a single integer on a line by itself, the number of admissible integer temperatures between a and b degrees, inclusive.

Examples

Input	Output
3 2 4 91 94 92 97 97 99 92 94 93 97 95 96 90 100	3 3 0 4

Input	Output
2 1 1 1 1 200000 200000 90 100	0

Note

In the first test case, Karen knows 3 recipes.

1. The first one recommends brewing the coffee between 91 and 94 degrees, inclusive.
2. The second one recommends brewing the coffee between 92 and 97 degrees, inclusive.
3. The third one recommends brewing the coffee between 97 and 99 degrees, inclusive.

A temperature is *admissible* if at least 2 recipes recommend it.

She asks 4 questions.

In her first question, she wants to know the number of admissible integer temperatures between 92 and 94 degrees, inclusive. There are 3: 92, 93 and 94 degrees are all admissible.

In her second question, she wants to know the number of admissible integer temperatures between 93 and 97 degrees, inclusive. There are 3: 93, 94 and 97 degrees are all admissible.

In her third question, she wants to know the number of admissible integer temperatures between 95 and 96 degrees, inclusive. There are none.

In her final question, she wants to know the number of admissible integer temperatures between 90 and 100 degrees, inclusive. There are 4: 92, 93, 94 and 97 degrees are all admissible.

In the second test case, Karen knows 2 recipes.

1. The first one, "wikiHow to make Cold Brew Coffee", recommends brewing the coffee at exactly 1 degree.
2. The second one, "What good is coffee that isn't brewed at at least 36.3306 times the temperature of the surface of the sun?", recommends brewing the coffee at exactly 200000 degrees.

A temperature is *admissible* if at least 1 recipe recommends it.

In her first and only question, she wants to know the number of admissible integer temperatures that are actually reasonable. There are none.

Problem F. Kuriyama Mirai's Stones

Time Limit	2000 ms
Mem Limit	262144 kB
Input File	stdin
Output File	stdout

Kuriyama Mirai has killed many monsters and got many (namely n) stones. She numbers the stones from 1 to n. The cost of the i-th stone is v_i. Kuriyama Mirai wants to know something about these stones so she will ask you two kinds of questions:

1. She will tell you two numbers, l and r ($1 \le l \le r \le n$), and you should tell her $\sum_{i=l}^{r} v_i$.
2. Let u_i be the cost of the i-th cheapest stone (the cost that will be on the i-th place if we arrange all the stone costs in non-decreasing order). This time she will tell you two numbers, l and r ($1 \le l \le r \le n$), and you should tell her $\sum_{i=l}^{r} u_i$.

For every question you should give the correct answer, or Kuriyama Mirai will say "fuyukai desu" and then become unhappy.

Input

The first line contains an integer n ($1 \le n \le 10^5$). The second line contains n integers: $v_1, v_2, \ldots, v_n$ ($1 \le v_i \le 10^9$) — costs of the stones.

The third line contains an integer m ($1 \le m \le 10^5$) — the number of Kuriyama Mirai's questions. Then follow m lines, each line contains three integers $type$, l and r ($1 \le l \le r \le n$; $1 \le type \le 2$), describing a question. If $type$ equal to 1, then you should output the answer for the first question, else you should output the answer for the second one.

Output

Print m lines. Each line must contain an integer — the answer to Kuriyama Mirai's question. Print the answers to the questions in the order of input.

Examples

Input	Output
6 6 4 2 7 2 7 3 2 3 6 1 3 4 1 1 6	24 9 28

Input	Output
4 5 5 2 3 10 1 2 4 2 1 4 1 1 1 2 1 4 2 1 2 1 1 1 1 3 3 1 1 3 1 4 4 1 2 2	10 15 5 15 5 5 2 12 3 5

Note

Please note that the answers to the questions may overflow 32-bit integer type.

Problem G. Frequency Queries

OS Linux

You are given q queries. Each query is of the form two integers described below:
- $1\ x$: Insert x in your data structure.
- $2\ y$: Delete one occurence of y from your data structure, if present.
- $3\ z$: Check if any integer is present whose frequency is exactly z. If yes, print 1 else 0.

The queries are given in the form of a 2-D array $queries$ of size q where $queries[i][0]$ contains the operation, and $queries[i][1]$ contains the data element.

Example
$queries = [(1,1),(2,2),(3,2),(1,1),(1,1),(2,1),(3,2)]$

The results of each operation are:

```
Operation   Array   Output
(1,1)       [1]
(2,2)       [1]
(3,2)                0
(1,1)       [1,1]
(1,1)       [1,1,1]
(2,1)       [1,1]
(3,2)                1
```

Return an array with the output: $[0,1]$.

Function Description

Complete the *freqQuery* function in the editor below.

freqQuery has the following parameter(s):

- *int queries[q][2]:* a 2-d array of integers

Returns
- *int[]:* the results of queries of type 3

Input Format

The first line contains of an integer q, the number of queries.
Each of the next q lines contains two space-separated integers, $queries[i][0]$ and $queries[i][1]$.

Constraints

- $1 \leq q \leq 10^5$
- $1 \leq x, y, z \leq 10^9$
- All $queries[i][0] \in \{1, 2, 3\}$
- $1 \leq queries[i][1] \leq 10^9$

Input	Output
8 1 5 1 6 3 2 1 10 1 10 1 6 2 5 3 2	0 1

Explanation 0

For the first query of type 3, there is no integer whose frequency is 2 ($array = [5, 6]$). So answer is 0.
For the second query of type 3, there are two integers in $array = [6, 10, 10, 6]$ whose frequency is 2 (integers = 6 and 10). So, the answer is 1.

Input	Output
4 3 4 2 1003 1 16 3 1	0 1

Explanation 1

For the first query of type 3, there is no integer of frequency 4. The answer is 0. For the second query of type 3, there is one integer, 16 of frequency 1 so the answer is 1.

Input	Output
10 1 3 2 3 3 2 1 4 1 5 1 5 1 4 3 2 2 4 3 2	0 1 1

Explanation 2

When the first output query is run, the array is empty. We insert two 4's and two 5's before the second output query, $arr = [4, 5, 5, 4]$ so there are two instances of elements occurring twice. We delete a 4 and run the same query. Now only the instances of 5 satisfy the query.

Problem H. Ilya and Queries

Time Limit	2000 ms
Mem Limit	262144 kB
Input File	stdin
Output File	stdout

Ilya the Lion wants to help all his friends with passing exams. They need to solve the following problem to pass the IT exam.

You've got string $s = s_1 s_2 \ldots s_n$ (n is the length of the string), consisting only of characters "." and "#" and m queries. Each query is described by a pair of integers l_i, r_i ($1 \le l_i < r_i \le n$). The answer to the query l_i, r_i is the number of such integers i ($l_i \le i < r_i$), that $s_i = s_{i+1}$.

Ilya the Lion wants to help his friends but is there anyone to help him? Help Ilya, solve the problem.

Input

The first line contains string s of length n ($2 \le n \le 10^5$). It is guaranteed that the given string only consists of characters "." and "#".

The next line contains integer m ($1 \le m \le 10^5$) — the number of queries. Each of the next m lines contains the description of the corresponding query. The i-th line contains integers l_i, r_i ($1 \le l_i < r_i \le n$).

Output

Print m integers — the answers to the queries in the order in which they are given in the input.

Examples

Input	Output
...... 4 3 4 2 3 1 6 2 6	1 1 5 4

Input	Output
#..### 5 1 3 5 6 1 5 3 6 3 4	1 1 2 2 0

Problem I. Cumulative Sum Query

Time Limit	7000 ms
Mem Limit	1572864 kB
Code Length Limit	50000 B
OS	Linux

William Macfarlane wants to look at an array.

You are given a list of **N** numbers and **Q** queries. Each query is specified by two numbers *i* and *j*; the answer to each query is the sum of every number between the range [i, j] (inclusive).

Note: the query ranges are specified using 0-based indexing.

Input

The first line contains **N**, the number of integers in our list (**N ≤ 100,000**). The next line holds **N** numbers that are guaranteed to fit inside an integer. Following the list is a number **Q** (**Q ≤ 10,000**). The next **Q** lines each contain two numbers **i** and **j** which specify a query you must answer (**0 ≤ i, j ≤ N-1**).

Output

For each query, output the answer to that query on its own line in the order the queries were made.

Example

Input	Output
3 1 4 1 3 1 1 1 2 0 2	4 5 6

Problem J. Frequency Array

Time Limit	1000 ms
Code Length Limit	50000 B
OS	Linux

A beautiful sequence is defined as a sequence that do not have any repeating elements in it.

You will be given any random sequence of integers, and you have to tell whether it is a beautiful sequence or not.

Input:

- The first line of the input contains a single integer T. T denoting the number of test cases. The description of T test cases is as follows.
- The next line of the input contains a single integer N. N denotes the total number of elements in the sequence.
- The next line of the input contains N space-separated integers $A1$, $A2$, $A3...An$ denoting the sequence.

Output:

- Print “prekrasnyy”(without quotes) if the given sequence is a beautiful sequence, else print “ne krasivo”(without quotes)

Note: each test case output must be printed on new line

Constraints:

- $1 \leq T \leq 10^2$
- $1 \leq N \leq 10^3$
- $1 \leq A1, A2, A3...An \leq 10^5$

Sample Input:

```
2
4
1 2 3 4
6
1 2 3 5 1 4
```

Sample Output:

```
prekrasnyy
ne krasivo
```

Explanation:

- As 1st sequence do not have any elements repeating, hence it is a beautiful sequence
- As in 2nd sequence the element 1 is repeated twice, hence it is not a beautiful sequence

Problem K. Beautiful Triplets

OS Linux

Given a sequence of integers a, a triplet $(a[i], a[j], a[k])$ is beautiful if:

- $i < j < k$
- $a[j] - a[i] = a[k] - a[j] = d$

Given an increasing sequenc of integers and the value of d, count the number of beautiful triplets in the sequence.

Example
$arr = [2, 2, 3, 4, 5]$
$d = 1$

There are three beautiful triplets, by index: $[i, j, k] = [0, 2, 3], [1, 2, 3], [2, 3, 4]$. To test the first triplet, $arr[j] - arr[i] = 3 - 2 = 1$ and $arr[k] - arr[j] = 4 - 3 = 1$.

Function Description

Complete the *beautifulTriplets* function in the editor below.

beautifulTriplets has the following parameters:

- *int d:* the value to match
- *int arr[n]:* the sequence, sorted ascending

Returns

- *int:* the number of beautiful triplets

Input Format

The first line contains 2 space-separated integers, n and d, the length of the sequence and the beautiful difference.
The second line contains n space-separated integers $arr[i]$.

Constraints

- $1 \leq n \leq 10^4$
- $1 \leq d \leq 20$
- $0 \leq arr[i] \leq 2 \times 10^4$
- $arr[i] > arr[i-1]$

Input	Output
STDIN ----- 7 3 1 2 4 5 7 8 10 Function -------- arr[] size n = 7, d = 3 arr = [1, 2, 4, 5, 7, 8, 10]	3

Explanation

There are many possible triplets $(arr[i], arr[j], arr[k])$, but our only beautiful triplets are $(1, 4, 7)$, $(4, 7, 10)$ and $(2, 5, 8)$ by value, not index. Please see the equations below:

$7 - 4 = 4 - 1 = 3 = d$
$10 - 7 = 7 - 4 = 3 = d$
$8 - 5 = 5 - 2 = 3 = d$

Recall that a beautiful triplet satisfies the following equivalence relation:
$arr[j] - arr[i] = arr[k] - arr[j] = d$ where $i < j < k$.