CS F211
Data Structures and Algorithms
Assignment - 6

Allowed languages: **C**

March 4, 2021

General Tips

- Try to use functions as much as possible in your code. Functions increase reusability and the pass-by-value feature provides a significant help sometimes. Modularizing your code also helps you to debug efficiently.
- Use `scanf` to read characters/strings from STDIN. Avoid using `getchar`, `getc` or `gets`. Try to read up about character suppression in `scanf` as it will be very helpful in some of the problems.
- Use `printf` instead of `putc`, `putchar` or `puts` to print character/string output on STDOUT.
- Indent your code appropriately and use proper variable names. These increase readability and writability of the code. Also, use comments wherever necessary.
- Use a proper IDEs like Sublime Text or VSCode as they help to run and test your code on multiple test-cases easily. You can install Windows Subsystem Linux (WSL) or MinGW 7.3.0, if you are Windows user to compile and run your programs. Alternatively, you can run and test your codes on Online GDB. If you are using WSL or Linux to run your programs, make sure that the gcc version is `gcc 5.4.1 c99`.

A: Disposition

Bob likes to watch the weather change so he wants to know today's forecast. He has a list of N different numbers representing a_i, the air pressure at various times of the day. Bob can tell the weather using only these numbers, but to do so he first needs to sort them in non-decreasing order based on the value of $a_i \mod k$, given k. For example, if $k = 5$ then the number 11 would come before 7 in the sorted array because $11 \mod 5$ is less than $7 \mod 5$. If the value of $a_i \mod k$ is equal to the value of $a_j \mod k$ for some i and j, then they must be sorted based on the values a_i and a_j (That is if $a_i < a_j$ then a_i comes before a_j). Sort the array in this way for Bob.

Input

The first line contains two integers N and k as described above ($1 \leq N \leq 10^5$, $1 \leq k \leq 10^5$). The second line of input contains N integers where the i^{th} number represents a_i ($1 \leq a_i \leq 10^9$).

Output

Print N integers in the sorted order as described in the question.

---

input
```
5 2
1 2 3 4 5
```

output
```
2 4 1 3 5
```

explanation

2 and 4 must come before the rest since they have a value $0 \mod 2$ and 1, 3, and 5 have a value $1 \mod 2$. Now 2 and 4 must be ordered based on their values and the same goes for $1, 3,$ and 5.

B: Reflection

It soon becomes night and Bob likes to talk to the moon before going to sleep. The moon tells him that the light is not her own but that of a thousand reflections passing over her. Bob noticed a row of N pine trees that was illuminated by the light, the i^{th} pine tree having a height h_i. He decided that he wants them to be sorted in non-decreasing order. To do this, he can divide the entire array of trees into contiguous segments such that each tree belongs to exactly one segment. He can then sort all the elements within each segment in non-decreasing order independently from the other segments. He must now choose segments such that after performing this operation, the entire array of trees is sorted in non-decreasing order. Bob doesn't want to waste energy on sorting so he wants to choose segments such that the number of segments chosen is as large as possible. Can you help Bob out?

Input

The first line contains an integer N representing the number of trees ($1 \leq N \leq 10^5$). The next line contains N numbers representing the height of the i^{th} tree h_i ($1 \leq h_i \leq 10^9$).

Output

Print a single integer, the largest possible number of segments you can choose satisfying the above conditions.

```
input
4
2 1 3 2

output
2
```

```
explanation
```

Here the optimal groups would be positions $1, 2$ in the first group and $3, 4$ in the second group. We can easily see that if we sort each segment individually the entire array will also be sorted.

C: Triad

Since this is the finale of Bob's trilogy, he decides to keep quiet and let the work be done by a bunch of machines. Bob is actually a winemaker, but he's a little weird and uses watermelons instead of grapes. He plans to use these machines to crush watermelons to use in his wine. There are N machines where the i^{th} machine takes time t_i to crush one watermelon. These machines can work simultaneously and independent of each other but one machine can only be crushing one watermelon at a time. You can assign a machine to begin crushing a watermelon at any time as long as the aforementioned rule is not violated. You can also use a machine more than once. Can you help Bob find the shortest amount of time needed to completely crush k watermelons?

Input

The first input line has two integers N ($1 \leq N \leq 10^5$) and k ($1 \leq k \leq 10^9$): the number of machines and watermelons. The next line has N integers where t_i ($1 \leq t_i \leq 10^9$) represents the time needed for the i^{th} machine to crush one watermelon.

Output

Output a single integer representing the shortest time needed to completely crush k watermelons.

```
input
3 7
3 2 5

output
8


explanation
The optimal way in this case would be for machine 1 to crush two watermelons, machine
2 to crush 4 watermelons and machine 3 to crush 1 watermelon.  This would take a total
time of 8
```

D: ANC Orders

The ANC burger stall, which serves only one dish - Jalapeño Poppers, is open for N minutes every night. **Each minute**, S_i tired and hungry students return from the library and attempt to order a plate of Jalapeño Poppers. To order food, the ANC employee takes each student's ID card and scans it with a barcode reader. The barcode scanner needs to be temporarily restarted in the $r_1^{th}, r_2^{th} ... r_k^{th}$ minute, and all students who arrive in these minutes will immediately leave (and will not come back) when they realise the barcode machine isn't working. The owner has a backup barcode scanner that works perfectly for exactly X minutes, without requiring these maintenance cycles, and then shut down permanently. What would be the optimal time to activate this backup device, if the owner wants to serve as many students as possible?

Input

The first line contains two integers N, k and X ($1 \leq k, X \leq N \leq 5 \times 10^6$), where N is the number of minutes the shop remains open and k is the number of minutes the barcode scanner doesn't work. The next line contains N space separated integers $S_0, S_1, \ldots S_{N-1}$ denoting the number of students arriving in the zeroeth, first $\ldots$ $(N-1)^{th}$ minute ($1 \leq S_i \leq 100$). The third line of input contains k (sorted) space separated integers $r_1, r_2 ... r_k$ meaning that the barcode scanner doesn't work in these k minutes. Note that time here is 0-indexed.

Output

Print two space separated integers - (1) how many students the shop serves without the extra barcode machine and (2) the *maximal* number of students the shop could serve if it optimally uses the second barcode machine.

```
input
5 2 1
1 0 1 2 1
0 3

output
2 4
```

explanation

Since the backup barcode scanner can work for one minute exactly, we can choose to use it at time, $t = 0$ or $t = 3$. We choose $t = 3$ since more students are coming in then.

E: Anomaly Detection

Anomalies are points that don't fit the "trend" of the rest of the data. Anomaly detection is the identification of rare items and points like these, representing that differ from the majority of the data by a lot. Rahul, a Data Scient PS2 intern at Macrobook, is attempting to implement a simple anomaly detection algorithm developed by research scientists at the company. Since Rahul has never written C code before, he approaches you for help. The algorithm is explained as follows:

You are given N data points each consisting of coordinates (x_i, y_i), where both coordinates are not integers (guaranteed to be floats). It is guaranteed that all N points will lie inside the square formed by the points (0, 0), (C,0), (A,D), (0,D), where C and D are two positive integers. We divide the X-axis into C intervals (or buckets) - namely x=0 to x=1, x=1 to x=2 and so on till x = C-1 to x = C. Similarly, the Y-axis is divided into D equally spaced buckets of unit size. These intervals form a 2D grid in the plane, with each point being located inside one X bucket and one Y bucket each - let us call these $B(x_i)$ and $B(y_i)$ respectively. The algorithm then assigns a normality score to each point $N_i = count(B(x_i)) * count(B(y_i))$, where $count(B)$ represents the number of points in that specific bucket. The k points with the least normality scores are anomalies and should be outputted. If there is a tie in scores, resolve them by outputting the one with lesser index in the original array.

Input

The first line consists of four space separated integers N $(1 \leq N \leq 10^7)$, C, D $(1 \leq C, D \leq 10^3)$, k $(1 \leq k \leq N)$. The next N lines contain two space separated floats each representing x_i and y_i $(0 < x_i < C, 0 < y_i < D)$ representing the i^{th} point, with i being zero-indexed.

Output

Print k space separated integers, representing the indices of the k anomalous points in non- order of normality scores.

```
input
4 10 10 2
3.1 3.2
3.3 7.7
9.1 3.2
0.1 2.9

output
3 1

explanation
The normality scores of the four points are 4, 2, 2, 1 respectively.  The least score
is of index 3's so we output that.  For the next highest we notice a tie between the
1st and 2nd index elements.  We output the lesser of these indices, i.e.  1
```

F: Array Operations (Again)

You are initially given an array A and a set of T instructions that you must evaluate sequentially, in order. Each instruction T is a number T_i - which means that you should rotate the array left by T_i places. After executing each instruction, print the first and last element of the resulting array.

Input

The first line consists of two space separated integers, N and T ($2 \leq T \leq 10^2$), where N ($0 < N < 10^4$) is the size of the array A ($0 \leq A_i \leq 10^9$). The second line contains N space separated integers, the elements of A. The third line of input contains T space separated integers representing the instructions ($0 \leq T_i \leq N$).

Output

Output T lines, each line containing two integers representing the first and last elements of the array after executing i^{th} query.

```
input
5 2
3 2 1 4 5
1 2

output
2 3
4 1

explanation
after the first operation, the array becomes 2 1 4 5 3.  After the second operation
the array becomes 4 5 3 2 1.
```

G: Maximal Descendent Distance

You are given a binary tree, consisting of **positive integer values** only, and each node is guaranteed to have a **unique** value. The binary tree is represented as an array A of size N. $A[0]$ is the root of the tree and for a given node i, $(2i + 1)$ and $(21 + 2)$ represent the left and right child respectively. If $A[i] = -1$ for any i, that means that the node i does not exist. Given two numbers a and b:

- find the location of both of these in the binary tree. If any one or both of these keys don't exist, print "-1".
- Identify two nodes a' and b', which are **leaf nodes** and are descendants of a and b respectively, such that the *hamming distance between values stored in nodes a' and b' respectively is minimized.*
- Print one integer, representing this minimal hamming distance.

Note: The hamming distance between two integers is defined as the number of bits that are different in the binary representations of the number. eg. Hamming distance between 3 (011) and 4 (100) is 3.

Note 2: Every node is considered to be a descendent of itself.

Input

The first line consists of three space separated integer N $(1 \leq N \leq 10^4)$, a, b. The next line consists of N space separated integers (A_i) representing the value of each node in the binary tree $(1 \leq A_i \leq 10^5$, or $A_i = -1)$.

Output

Output one integer, the minimum hamming distance, as described in the problem.

```
input
7 2 3
1 2 3 -1 -1 4 5

output
2

explanation
The only possible leaf-descendent of 2 is 2.  The possible leaf-descendants of 3 are
4 and 5.  Hamming distance of (2, 4) = 2 and hamming distance of (2, 5) = 3.  Minimal
of these distances is the answer.
```

H: Special String Activities

You are given a string consisting of alphabets and brackets "()[]". It is guaranteed that the brackets are balanced. You need to process the string by:

- Reverse substrings that are located within "()". For eg. $(aksj)$ becomes $jska$ after processing. (brackets removed).
- Increment all characters located within "[]". For eg. $[abcd]$ becomes $bcde$ after processing (brackets removed). The character z would become a after incrementing.
- Process the brackets from innermost bracket to outermost bracket.

Print the processed string.

Input

The first line contains a string as described in the question above. It is guaranteed that the length of the string $\leq 10^4$.

Output

Output one line, consisting of only lowercase English alphabets, denoting the processed string.

```
input
abcdegfg
output
abcdegfg

input
a(bc)deg[fg]
output
acbdeggh

input
a(bc[de]gf)g
output
afgfecbg

input
a[[[b]]]bujshs((dg))
output
aebujshsdg
```

I: All Might vs The League of Villains

The league of villains was finally able to corner the great hero All Might. Underestimating his valor, they all decide to charge at him at once. As All Might's sidekick, you (Dave) see the locations of all the villains and All Might as points (with X and Y coordinates) on your radar. You know that the bottom-most point (the minimum y-coordinate) on the radar corresponds to that of All Might. You recommend to fend off the villains in a in a single counter clockwise sweep (see sample case for clarity). Help All Might defend himself against the league of villains in a counter clockwise manner. If more than one villain has the same orientation with respect to All Might, you suggest All Might to first fend off the villain who is closer to him (in terms of manhattan distance). Assume that All Might does not move from his point.

Input

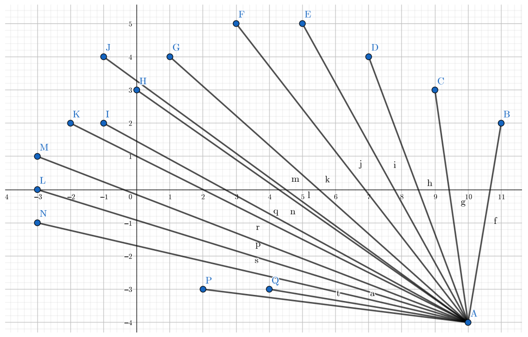
The first line of input contains a single integer N ($2 \leq \text{N} \leq 10^5$) denoting the number of points on your radar. Each of the following N lines contain three space-separated integers P, X_P, Y_P ($1 \leq \text{P} \leq \text{N}$, $-10^9 \leq X_P$, $Y_P \leq 10^9$) denoting an index number and its corresponding coordinates as you see on your radar. It is guaranteed the coordinates given will result in a unique bottom-most point.

Output

Print a sequence of N-1 integers (B), where B_i denotes the index of the villain he fends off in the i^{th} instant (in counter-clockwise direction). Figure for sample testcase-1 is given below.

```
input
16
12 -3 -1
15 5 5
13 2 -3
14 3 5
1 11 2
2 -1 4
10 -2 2
5 7 4
4 9 3
7 -3 1
6 -3 0
8 10 -4
9 -1 2
16 1 4
3 4 -3
11 0 3

output
1 4 5 15 14 16 2 11 9 10 7 6 12 3 13
```



J. Cube-Root

In this problem, you need to find the integral part of the cube-root of a given integer without using any inbuilt math functions. Further, your solution must run in o(N) time.

Input

The only line of input contains a single integer N ($-10^9 \leq N \leq 10^9$) for which the cube-root has to be computed.

Output

Print a single integer X denoting the integral part of the cube-root of the given integer N.

```
input
-125

output
-5
```

```
input
2348247

output
132
```

```
input
-617491

output
-85
```
