

## Problem A. Donut

Input file: *standard input*  
Output file: *standard output*  
Time limit: 3 seconds  
Memory limit: 1024 mebibytes

In this problem, we will use Chebyshev distance on a Cartesian plane. The Chebyshev distance between two points  $(p_x, p_y)$  and  $(q_x, q_y)$  on the plane is  $\max(|p_x - q_x|, |p_y - q_y|)$ .

Let us define a **donut** on the plane as the set of points in a certain distance range from a certain point, the donut's center. More formally, the donut with center  $(x_c, y_c)$ , inner radius  $l$  and outer radius  $r$  is the set of points  $(x, y)$  such that  $l \leq \max(|x - x_c|, |y - y_c|) \leq r$ .

There are  $n$  points selected on the Cartesian plane. Points are numbered from 1 to  $n$ , and each point has a score associated with it.

We want to place a donut on the plane. The inner radius of the donut will be  $l$ , the outer radius will be  $r$ , and the center of the donut is not yet determined: you have to decide where to place it. However, both coordinates of the center must be integers. After you place the donut, its score will be the sum of the scores of points which belong to the donut.

You are given the coordinates of the selected points and the score associated with each point. Calculate the maximum possible score of the donut you can place.

### Input

The first line contains three integers:  $n$ , the number of selected points on the plane,  $l$ , the inner radius of the donut, and  $r$ , the outer radius of the donut ( $1 \leq n \leq 10^5$ ,  $1 \leq l \leq r \leq 10^9$ ).

The following  $n$  lines describes selected points, one per line. Each of these lines contains three integers  $x$ ,  $y$ , and  $s$ : the coordinates of the point and the score associated with it ( $-10^9 \leq x, y \leq 10^9$ ,  $-10^4 \leq s \leq 10^4$ ). Note that some points may coincide.

### Output

Print the maximum score of the donut.

### Examples

standard input	standard output
4 1 1 0 1 1 0 -1 1 1 0 -100 -1 0 -100	2
4 1 2 0 1 1 0 -1 1 1 0 -100 -1 0 -100	1

## Problem B. Circular Arrangement

Input file: *standard input*  
Output file: *standard output*  
Time limit: 1 second  
Memory limit: 1024 mebibytes

Arraylover Zigui has  $n$  positive integers  $c_1, c_2, \dots, c_n$ . He made every possible array that contains exactly  $c_i$  copies of integer  $i$  for each  $i$  from 1 to  $n$ . For example, when  $n = 3$ ,  $c_1 = 4$ ,  $c_2 = 3$ , and  $c_3 = 2$ , he have made many arrays, one of which is the array  $[1, 1, 3, 3, 1, 2, 2, 2, 1]$ .

Koo is a big fan of Zigui. He wants to make the same arrays as Zigui. But unlike Zigui, he loves **circular** things. Thus, he will make every array circular. In a circular array, the first element and the last element are adjacent.

Making an array has a certain cost. More specifically, the cost of making an array is the product of sizes of adjacent groups of equal values. For example, the array  $[1, 1, 3, 3, 1, 2, 2, 2, 1]$  contains five adjacent groups of equal values: two 1s, then two 3s, then one 1, then three 2s, and finally, one 1. The cost of making this array is therefore  $2 \times 2 \times 1 \times 3 \times 1 = 12$ .

In a circular array, if the values of the first element and the last element are equal, their groups are merged. Thus, the cost of making a circular array  $[1, 1, 3, 3, 1, 2, 2, 2, 1]$  is  $(2 + 1) \times 2 \times 1 \times 3 = 18$ .

Calculate the sum of the costs of all circular arrays Koo will make. Note that, in a circular array, the index of each element is still important, just like in a regular array. So, for example,  $[1, 2, 1, 2]$  and  $[2, 1, 2, 1]$  are different circular arrays. As the total cost may be very large, calculate this sum modulo  $10^9 + 7$ .

### Input

The first line contains  $n$ , the number of distinct integers to use ( $2 \leq n \leq 50$ ).

The second line contains  $n$  positive integers  $c_1, c_2, \dots, c_n$ , where  $c_i$  is the number of occurrences of integer  $i$  in each desired array ( $1 \leq c_i \leq 100$ ).

### Output

Print the sum of costs of all of all circular arrays Koo will make, modulo  $10^9 + 7$ .

### Examples

standard input	standard output
2 2 2	18
3 4 3 2	7830
4 4 4 4 4	818559048
5 1 2 3 4 5	342934740
6 7 8 9 10 11 12	609539975

### Note

For the first example, we can make six circular arrays. Here are the arrays and their costs:

$[1, 1, 2, 2]$ : 4	$[1, 2, 1, 2]$ : 1	$[1, 2, 2, 1]$ : 4
$[2, 1, 1, 2]$ : 4	$[2, 1, 2, 1]$ : 1	$[2, 2, 1, 1]$ : 4

The sum of all the costs is 18.

## Problem C. Earthquake

Input file: *standard input*  
Output file: *standard output*  
Time limit: 1 second  
Memory limit: 512 mebibytes

The Island of Sodor (or just Sodor) can be reached from England via bridges.

Because Sodor and England are far away, there exist multiple routes (exactly  $n$  routes) that connect the two lands via one or more bridges. Specifically, route  $i$  connects Sodor and England through  $k_i$  ( $k_i \geq 1$ ) bridges. Let us denote the  $j$ -th bridge (counting from Sodor to England) of route  $i$  as  $B[i, j]$ . Below, we have two routes ( $n = 2$ ) and five bridges with  $k_1 = 2$  and  $k_2 = 3$ .

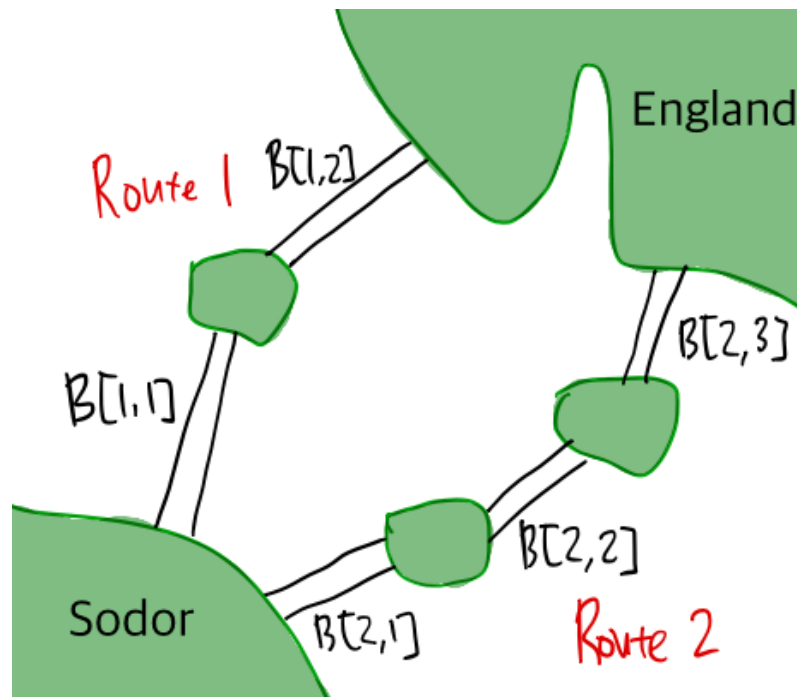


Figure 1: Two routes and five bridges

For any two adjacent bridges of a route,  $B[i, j]$  and  $B[i, j + 1]$  where  $j + 1 \leq k_i$ , their joint is a small island that simply serves as a connecting point of the two bridges. As demonstrated in the figure, there are exactly  $n$  routes that connect Sodor and England, as the bridges do not intersect, and all joints are also distinct. In particular, if any one of the bridges of route  $i$  is damaged, then route  $i$  cannot be used to travel between Sodor and England.

Due to recent earthquakes in the area, some of the bridges may have experienced severe damage and become unusable. At this point we do not know exactly which bridges withstood the earthquakes and which ones were destroyed. Thanks to the inspections that were performed on the bridges prior to the incidents, for each bridge, we know the exact probability of whether it is still intact or not. Let  $p[i, j]$  be the probability ( $0 < p[i, j] < 1$ ) that bridge  $B[i, j]$  is still intact after the earthquakes. Assume that the events of bridges being intact are mutually independent.

We want to know whether there is still a path between Sodor and England. However, determining whether a bridge is still intact or not is a costly operation because one needs to dispatch a large team of inspectors via choppers and boats, and therefore we want to minimize the number of inspections. Using an optimal sequence of inspections (that minimizes the expected number of inspections), what is the expected number of inspections we must perform until we are certain whether there is still a safe route between Sodor and England?

## Input

The first line contains  $n$ , the number of routes ( $2 \leq n \leq 1000$ ).

The following  $n$  lines contain the information on each route. The  $i$ -th of them starts with an integer  $k_i$ , the number of bridges in the  $i$ -th route ( $1 \leq k_i \leq 1000$ ). Then follow  $k_i$  integers labeled as  $q[i, j]$ , where  $1 \leq j \leq k_i$ . Each  $q[i, j]$  is an integer between 1 and 999, inclusive, and it means that  $p[i, j] = q[i, j]/1000$ .

## Output

Simply output the expected number of inspections for an optimal inspection sequence of the bridges. Your answer will be considered correct if its relative or absolute error is within  $10^{-9}$ .

## Examples

standard input	standard output
2 3 900 900 900 2 100 100	3.0081
3 1 240 1 310 1 50	2.2144

## Note

The first input describes the example discussed in the problem description. Intuitively, route 1 is very likely to be fine, as all three bridges are expected to be intact, whereas route 2 is most likely destroyed. The optimal sequence is to inspect the three bridges of route 1 (in any order, until either route 1 is declared to be safe or damaged), and then the two bridges of route 2.

For the second input, it is optimal to inspect route 2, then route 1, and then route 3 (if needed).

## Problem D. Dynamic Input Tool

Input file: *standard input*  
Output file: *standard output*  
Time limit: 1 second  
Memory limit: 512 mebibytes

CodingSlave 1.0 is a brand new text editor. There are two possible operations to compose a string in CodingSlave 1.0:

1. Add one character at the end of the current string.
2. Choose a nonempty subsequence of the current string and add it at the end of the current string.

Initially, the current string is empty.

Given a word consisting of lowercase English letters, calculate the minimum number of operations needed to compose this string.

### Input

The first line contains a string  $S$  consisting of lowercase English letters ( $1 \leq |S| \leq 10^6$ ).

### Output

Print one integer: the minimum number of operations needed to compose  $S$  using CodingSlave 1.0.

### Examples

<i>standard input</i>	<i>standard output</i>
aaa	3
aabaaaabaa	5

## Problem E. Central Lake

Input file: *standard input*  
Output file: *standard output*  
Time limit: 5 seconds  
Memory limit: 512 mebibytes

The town where Jaehyun lives is a circle with a radius of  $R$ . There are 360 000 points on the circumference, numbered in counterclockwise order: if you go counterclockwise from point 359 999, the next point is point 0. All the distances between pairs of neighboring points are equal.

Initially, there were  $N$  houses in some of the 360 000 points. Because the entire territory was originally flat, people could always go to each other's house by direct path. However, Sunghyeon, the mayor of the city, ordered to dig a lake in the center of the town to make it look good. The center of the lake and the center of the town is the same point. He also plans to demolish old houses and build new ones.

Jaehyun is worried that it will take a long time to walk between the houses because of the central lake. Your task is to calculate the maximum value of the shortest distance between two houses. Of course, every time the mayor orders to build or demolish a house, you have to recalculate the answer.

### Input

The first line of input contains two space-separated integers  $R$  and  $r$ : the radius of the country and the radius of the lake, respectively ( $10 \leq R \leq 10^5$ ,  $1 \leq r < R$ ).

The second line contains an integer  $N$  ( $2 \leq N \leq 100\,000$ ). The third line contains  $N$  space-separated integers  $a_1, a_2, \dots, a_N$ : the locations of  $N$  houses ( $0 \leq a_i < 360\,000$ , all  $a_i$  are pairwise distinct).

The next line contains an integer  $Q$ , the number of queries ( $1 \leq Q \leq 100\,000$ ).

The following  $Q$  lines describe the queries. Each of these lines contains two space-separated integers  $q$  and  $x$  ( $1 \leq q \leq 2$ ,  $0 \leq x < 360\,000$ ).

Each query has one of following formats depending on its type:

- "1  $x$ ": Build new house at point  $x$ .
- "2  $x$ ": Demolish a house at point  $x$ .

It is guaranteed that there is no house at point  $x$  when  $q$  is 1 and there is a house at point  $x$  when  $q$  is 2. Also, it is guaranteed that there will be at least two houses at all times.

### Output

Print  $Q + 1$  lines. On the first line, print the maximum distance between two houses after the lake appears, but before all queries. For the next  $Q$  lines, output the required answer after executing each query. Absolute or relative error  $10^{-6}$  or better will be tolerated.

### Example

standard input	standard output
10 5	14.14213562
2	22.55649583
0 90000	22.55649583
4	19.93850195
1 180000	10
1 240000	
2 0	
2 90000	

## Problem F. Computing MDSST

Input file: *standard input*  
Output file: *standard output*  
Time limit: 4 seconds  
Memory limit: 512 mebibytes

In this problem, we will consider weighted undirected graphs where all edges have positive weights.

Let  $D(G, i, j)$  be the length of the shortest path in graph  $G$  between vertex  $i$  and vertex  $j$ .

We are given a complete weighted undirected graph  $G$  which consists of  $n$  vertices numbered from 1 to  $n$ . Among the spanning trees of  $G$ , the MDSST (Minimum Distance Sum Spanning Tree) is such  $T$  for which the value  $S(T) = \sum_{1 \leq i < j \leq n} D(T, i, j)$  is minimum. Your task is to find MDSST of  $G$  and print  $S(T)$ .

### Input

The first line contains an integer  $n$ , the number of vertices in the graph ( $2 \leq n \leq 15$ ). The  $i$ -th of the following  $n - 1$  lines contains  $n - i$  integers separated by spaces. The  $j$ -th integer of this the line is the length of the edge between vertex  $i$  and vertex  $i + j$ .

All the lengths are between 1 and  $10^9$ , inclusive.

### Output

On the first line, print one integer: the value  $S(T)$  for the MDSST you found.

### Example

standard input	standard output
4 3 2 1 5 6 7	18



## Problem G. MST with Metropolis

Input file: *standard input*  
Output file: *standard output*  
Time limit: 5 seconds  
Memory limit: 1024 mebibytes

In this problem, we will consider weighted undirected graphs where all edges have positive weights.

We are given a weighted undirected graph  $G$  which consists of  $n$  vertices numbered from 1 to  $n$ . Among the spanning trees of  $G$ , the MST (Minimum Spanning Tree) with **metropolis vertex**  $i$  is the one that contains every possible edge with the metropolis (the vertex  $i$ ) and minimizes the sum of edge weights in it. Let this edge weight sum be  $S_i$ . Your task is to calculate  $S_i$  for every vertex  $i$ .

### Input

The first line contains two integers  $n$  and  $m$ : the number of vertices and edge in the graph, respectively ( $1 \leq n \leq 10^5$ ,  $n - 1 \leq m \leq 3 \times 10^5$ ).

Each of the following  $m$  lines describes a single edge and contains three integers,  $x$ ,  $y$ , and  $c$  which mean that there is an edge between vertices  $x$  and  $y$  of weight  $c$  ( $1 \leq x < y \leq n$ ,  $1 \leq c \leq 10^9$ ).

It is guaranteed that the given graph is connected, and there is at most one edge between every possible pair of vertices.

### Output

Print  $n$  lines. The  $i$ -th line must contain an integer  $S_i$ : the weight of the MST with metropolis vertex  $i$ .

### Example

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

## Problem H. Number of Cycles

Input file: *standard input*  
 Output file: *standard output*  
 Time limit: 3 seconds  
 Memory limit: 512 mebibytes

Jaehyun likes computational geometry. Here is Jaehyun's question: "We are given  $n$  segments on the Cartesian plane. Count the number of simple cycles in the generated graph."

Formally, a set of  $n$  segments  $S = \{s_1, s_2, \dots, s_n\}$  generates the following graph  $G = (V, E)$ .

For a point  $v$  of the plane,  $v \in V$  if  $v$  is one of the endpoints of the segments or  $v$  is an intersection of two or more segments.

For two distinct vertices  $u$  and  $v$ ,  $(u, v) \in E$  if there is a segment  $s_i \in S$  containing vertices  $u$  and  $v$ , and there is no vertex on  $s_i$  between  $u$  and  $v$ .

A simple cycle is a cycle with no repeated vertices or edges.

Zigui tried to solve Jaehyun's problem, and he found it is possible to make various answers with just a few segments.

For given  $N$ , find a set of segments such that the number of simple cycles in the graph generated by these segments is  $N$ .

### Input

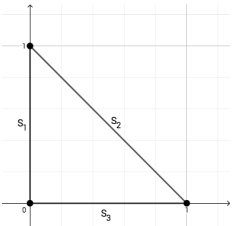
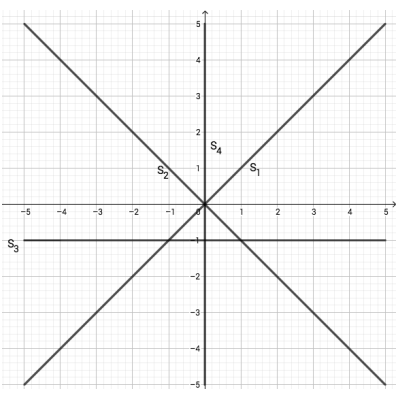
The first line contains an integer  $N$  ( $1 \leq N \leq 1000$ ).

### Output

The first line of the output must contain an integer  $K$ : the number of segments ( $1 \leq K \leq 12$ ).

Each of the next  $K$  lines must contain four integers  $x_1, y_1, x_2,$  and  $y_2$  denoting a segment with endpoints  $(x_1, y_1)$  and  $(x_2, y_2)$  ( $-10^9 \leq x_1, y_1, x_2, y_2 \leq 10^9, (x_1, y_1) \neq (x_2, y_2)$ ).

### Examples

standard input	standard output	Notes
1	3 0 0 0 1 1 0 0 1 1 0 0 0	
3	4 -5 -5 5 5 -5 5 5 -5 -5 -1 5 -1 0 -5 0 5	

## Problem I. Sum of Squares of the Occurrence Counts

Input file: *standard input*  
Output file: *standard output*  
Time limit: 2 seconds  
Memory limit: 1024 mebibytes

For two strings  $S$  and  $P$ , let the value  $f(S, P)$  be the number of occurrences of  $P$  in  $S$ . For example,  $f(\text{"ababa"}, \text{"aba"}) = 2$ , and  $f(\text{"aaaaa"}, \text{"aa"}) = 4$ .

For a string  $S$ , let the value  $g(S)$  be the sum of squares of the occurrence counts in  $S$  for all possible non-empty strings. More formally,  $g(S) = \sum (f(S, P)^2)$  for all possible non-empty strings  $P$ .

You have given a string of length  $n$ :  $S = c_1c_2 \dots c_n$ . Let  $r_i = g(c_1c_2 \dots c_i)$ . Calculate the values  $r_i$  for all  $i$  from 1 to  $n$ .

### Input

The first line contains the string  $S$  which consists of lowercase English letters. The length of  $S$  is between 1 and  $10^5$  letters, inclusive.

### Output

Print  $n$  lines. The  $i$ -th line must contain the integer  $r_i$ .

### Examples

standard input	standard output
aaa	1 5 14
pqpq	1 3 8 16
stssts	1 3 8 16 25 41

## Problem J. Game of Sorting

Input file: *standard input*  
Output file: *standard output*  
Time limit: 3 seconds  
Memory limit: 512 mebibytes

Alice and Bob have invented a new game to play. First, they get a sequence. And then they take turns to make the following moves. During each move, the player will choose either the first element of the sequence or the last element, and remove the chosen element. The player who makes the sequence nondecreasing or nonincreasing wins. If the initial sequence is a nondecreasing or nonincreasing sequence, Bob wins the game.

The winter vacation is boring, so the kids want to play this game many times. Initially, they have a sequence of length  $n$ :  $a_1, a_2, \dots, a_n$ . Alice realized that if they start the game after removing some of the first elements of the sequence and some of its last elements, you can get completely different results.

Alice and Bob played the game  $Q$  times in total. The question is who will finally win each game if both players play optimally. Remember that Alice always moves first.

### Input

The first line contains an integer  $n$ , the length of the initial sequence ( $3 \leq n \leq 10^6$ ).

The second line contains  $n$  space-separated integers  $a_1, a_2, \dots, a_n$ : the sequence itself ( $1 \leq a_i \leq 10^9$ ).

The third line contains an integer  $Q$  ( $1 \leq Q \leq 10^6$ ).

The  $i$ -th of the following  $Q$  lines contains integers  $L_i$  and  $R_i$  ( $1 \leq L_i \leq R_i \leq n$ ). It means that the initial sequence of  $i$ -th game is  $a_{L_i}, a_{L_i+1}, \dots, a_{R_i}$ .

### Output

Print  $Q$  lines with the winner's name, one for each query.

### Example

standard input	standard output
4	Bob
1 5 3 4	Alice
3	Bob
1 2	
1 3	
1 4	

## Problem K. Subsequence Queries

Input file: *standard input*  
Output file: *standard output*  
Time limit: 5 seconds  
Memory limit: 1024 mebibytes

A **subsequence** of string  $S$  is a string that can be obtained from  $S$  by removing zero or more characters without changing the order of the remaining characters. For example, the string “ababA” has 24 different subsequences: (empty string), “a”, “b”, “A”, “ab”, “aa”, “aA”, “ba”, “bb”, “bA”, “aba”, “abb”, “abA”, “aab”, “aaA”, “bab”, “baA”, “bbA”, “abab”, “abaA”, “abbA”, “aabA”, “baba”, and “ababA”. Note that we distinguish uppercase and lowercase letters in this problem.

You are given a string of length  $n$ ,  $S = c_1c_2\dots c_n$ , and  $Q$  contiguous substrings of  $S$ . Calculate the number of different subsequences for each of these  $Q$  substrings.

To reduce and encode input, we will use the following pseudo-random generator. The numbers  $a_0$ ,  $b_0$ ,  $p$ ,  $q$ , and  $r$  are given initially. The values  $a_i$ ,  $b_i$ ,  $x_i$ ,  $y_i$ ,  $z_i$  are then produced as follows (let  $z_0 = 0$  and  $X = 10^9 + 7$  for convenience):

$$\begin{aligned}a_i &= (p \cdot a_{i-1} + q \cdot b_{i-1} + z_{i-1} + r) \bmod X \\b_i &= (p \cdot b_{i-1} + q \cdot a_{i-1} + z_{i-1} + r) \bmod X \\x_i &= \min(a_i \bmod n, b_i \bmod n) + 1 \\y_i &= \max(a_i \bmod n, b_i \bmod n) + 1 \\z_i &= (\text{the number of different subsequences of } c_{x_i}c_{x_i+1}\dots c_{y_i-1}c_{y_i})\end{aligned}$$

Your program must print only one integer: the value  $z_Q$ .

### Input

The first line contains the string  $S$  which can contain uppercase and lowercase English letters. The length of  $S$  is between 1 and  $10^6$  letters, inclusive.

The second line contains six integers  $Q$ ,  $a_0$ ,  $b_0$ ,  $p$ ,  $q$ , and  $r$  ( $1 \leq Q \leq 10^6$ ,  $0 \leq a_0, b_0, p, q, r < X = 10^9 + 7$ ).

### Output

Print the integer  $z_Q$ .

### Example

standard input	standard output
ababA 2 4 6 8 10 12	7

## Problem L. XOR Transformation

Input file: *standard input*  
 Output file: *standard output*  
 Time limit: 3 seconds  
 Memory limit: 1024 mebibytes

You are given an integer array of length  $N$ :  $X = [x_0, x_1, \dots, x_{N-1}]$ .

Let us define a transformation of  $X$ , which is  $F_k(X) = [f_{k,0}(X), f_{k,1}(X), \dots, f_{k,N-1}(X)]$ , as follows:

- $k$  is an integer between 1 and  $N$ , inclusive.
- $f_{k,i}(X) = \bigoplus_{j=0}^{k-1} x_{(i+j) \bmod N}$ , where  $i$  is an integer between 0 and  $N-1$ , inclusive, and  $\oplus$  is bitwise XOR.

You are also given two integers  $T$  and  $K$ . Calculate the value  $F_K^T(X)$  and print it. Note that  $F_K^1(X) = F_K(X)$  and  $F_K^t(X) = F_K(F_K^{t-1}(X))$  for  $t > 1$ .

### Input

The first line contains three integers:  $N$ ,  $K$ , and  $T$  ( $1 \leq K \leq N \leq 10^5$ ,  $1 \leq T \leq 10^{18}$ ).

The second line contains  $N$  non-negative integers  $x_0, x_1, \dots, x_{N-1}$  ( $0 \leq x_i \leq 10^9$ ), where  $x_i$  is the  $i$ -th element of the array  $X$ , numbered from zero.

### Output

Let  $F_K^T(X) = [a_0, a_1, \dots, a_{N-1}]$ . Print the  $N$  integers  $a_0, a_1, \dots, a_{N-1}$  on the first line.

### Examples

standard input	standard output
5 3 1 3 0 2 1 2	1 3 1 0 1
5 3 2 3 0 2 1 2	3 2 0 0 3
5 3 3 3 0 2 1 2	1 2 3 0 2
5 3 15 3 0 2 1 2	3 0 2 1 2
11 5 10000000000000000000 2 2 4 5 9 1 5 7 7 1 8	13 4 5 8 1 0 5 10 3 4 8