Educational Codeforces Round 48 (Rated for Div. 2)

A. Death Note

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You received a notebook which is called *Death Note*. This notebook has infinite number of pages. A rule is written on the last page (huh) of this notebook. It says: "You have to write names in this notebook during n consecutive days. During the i-th day you have to write exactly a_i names.". You got scared (of course you got scared, who wouldn't get scared if he just receive a notebook which is named *Death Note* with a some strange rule written in it?).

Of course, you decided to follow this rule. When you calmed down, you came up with a strategy how you will write names in the notebook. You have calculated that each page of the notebook can contain exactly m names. You will start writing names from the first page. You will write names on the current page as long as the limit on the number of names on this page is not exceeded. When the current page is over, you turn the page. Note that you *always* turn the page when it ends, it doesn't matter if it is the last day or not. If after some day the current page still can hold at least one name, during the next day you will continue writing the names from the current page.

Now you are interested in the following question: how many times will you turn the page during each day? You are interested in the number of pages you will turn each day from 1 to n.

Input

The first line of the input contains two integers n, m ($1 \le n \le 2 \cdot 10^5$, $1 \le m \le 10^9$) — the number of days you will write names in the notebook and the number of names which can be written on each page of the notebook.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^9$), where a_i means the number of names you will write in the notebook during the i-th day.

Output

Print exactly n integers $t_1, t_2, \dots, t_n$, where t_i is the number of times you will turn the page during the i-th day.

Examples

input
```
3 5
3 7 9
```
output
```
0 2 1
```
input
```
4 20
10 9 19 2
```
output
```
0 0 1 1
```
input
```
1 100
99
```
output
```
0
```

Note

In the first example pages of the *Death Note* will look like this $[1, 1, 1, 2, 2], [2, 2, 2, 2, 2], [3, 3, 3, 3, 3], [3, 3, 3, 3]$. Each number of the array describes during which day name on the corresponding position will be written. It is easy to see that you should turn the first and the second page during the second day and the third page during the third day.

B. Segment Occurrences

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given two strings s and t, both consisting only of lowercase Latin letters.

The substring $s[l..r]$ is the string which is obtained by taking characters $s_l, s_{l+1}, \dots, s_r$ without changing the order.

Each of the occurrences of string a in a string b is a position i ($1 \le i \le |b| - |a| + 1$) such that $b[i..i + |a| - 1] = a$ ($|a|$ is the length of string a).

You are asked q queries: for the i-th query you are required to calculate the number of occurrences of string t in a substring $s[l_i..r_i]$.

Input

The first line contains three integer numbers n, m and q ($1 \le n, m \le 10^3$, $1 \le q \le 10^5$) — the length of string s, the length of string t and the number of queries, respectively.

The second line is a string s ($|s| = n$), consisting only of lowercase Latin letters.

The third line is a string t ($|t| = m$), consisting only of lowercase Latin letters.

Each of the next q lines contains two integer numbers l_i and r_i ($1 \le l_i \le r_i \le n$) — the arguments for the i-th query.

Output

Print q lines — the i-th line should contain the answer to the i-th query, that is the number of occurrences of string t in a substring $s[l_i..r_i]$.

Examples

input

```
10 3 4
codeforces
for
1 3
3 10
5 6
5 7
```

output

```
0
1
0
1
```

input

```
15 2 3
abacabadabacaba
ba
1 15
3 4
2 14
```

output

```
4
0
3
```

input

```
3 5 2
aaa
baaab
1 3
1 1
```

output

```
0
0
```

Note

In the first example the queries are substrings: "`cod`", "`deforces`", "`fo`" and "`for`", respectively.

C. Vasya And The Mushrooms

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Vasya's house is situated in a forest, and there is a mushroom glade near it. The glade consists of two rows, each of which can be divided into n consecutive cells. For each cell Vasya knows how fast the mushrooms grow in this cell (more formally, how many grams of mushrooms grow in this cell each minute). Vasya spends exactly one minute to move to some adjacent cell. Vasya cannot leave the glade. Two cells are considered adjacent if they share a common side. When Vasya enters some cell, he instantly collects all the mushrooms growing there.

Vasya begins his journey in the left upper cell. Every minute Vasya must move to some adjacent cell, he cannot wait for the mushrooms to grow. He wants to visit all the cells **exactly once** and maximize the total weight of the collected mushrooms. Initially, all mushrooms have a weight of 0. Note that Vasya doesn't need to return to the starting cell.

Help Vasya! Calculate the maximum total weight of mushrooms he can collect.

Input

The first line contains the number n $(1 \le n \le 3 \cdot 10^5)$ — the length of the glade.

The second line contains n numbers $a_1, a_2, \dots, a_n$ $(1 \le a_i \le 10^6)$ — the growth rate of mushrooms in the first row of the glade.

The third line contains n numbers $b_1, b_2, \dots, b_n$ $(1 \le b_i \le 10^6)$ is the growth rate of mushrooms in the second row of the glade.

Output

Output one number — the maximum total weight of mushrooms that Vasya can collect by choosing the optimal route. Pay attention that Vasya must visit every cell of the glade **exactly once**.

Examples

input

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

output

```
70
```

input

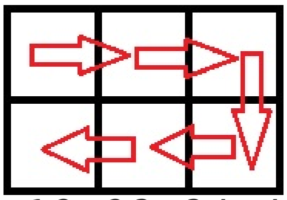
```
3
1 1000 10000
10 100 100000
```

output

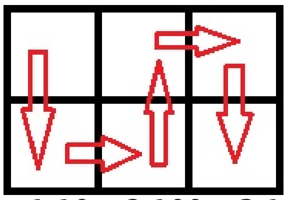
```
543210
```

Note

In the first test case, the optimal route is as follows:



Thus, the collected weight of mushrooms will be $0 \cdot 1 + 1 \cdot 2 + 2 \cdot 3 + 3 \cdot 4 + 4 \cdot 5 + 5 \cdot 6 = 70$.
In the second test case, the optimal route is as follows:



Thus, the collected weight of mushrooms will be $0 \cdot 1 + 1 \cdot 10 + 2 \cdot 100 + 3 \cdot 1000 + 4 \cdot 10000 + 5 \cdot 100000 = 543210$.

D. Vasya And The Matrix

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Now Vasya is taking an exam in mathematics. In order to get a good mark, Vasya needs to guess the matrix that the teacher has constructed!

Vasya knows that the matrix consists of n rows and m columns. For each row, he knows the xor (bitwise excluding or) of the elements in this row. The sequence $a_1, a_2, \dots, a_n$ denotes the xor of elements in rows with indices $1, 2, \dots, n$, respectively. Similarly, for each column, he knows the xor of the elements in this column. The sequence $b_1, b_2, \dots, b_m$ denotes the xor of elements in columns with indices $1, 2, \dots, m$, respectively.

Help Vasya! Find a matrix satisfying the given constraints or tell him that there is no suitable matrix.

Input

The first line contains two numbers n and m $(2 \le n, m \le 100)$ — the dimensions of the matrix.

The second line contains n numbers $a_1, a_2, \dots, a_n$ $(0 \le a_i \le 10^9)$, where a_i is the xor of all elements in row i.

The third line contains m numbers $b_1, b_2, \dots, b_m$ $(0 \le b_i \le 10^9)$, where b_i is the xor of all elements in column i.

Output

If there is no matrix satisfying the given constraints in the first line, output "NO".

Otherwise, on the first line output "YES", and then n rows of m numbers in each $c_{i1}, c_{i2}, \ldots, c_{im}$ $(0 \le c_{ij} \le 2 \cdot 10^9)$ — the description of the matrix.

If there are several suitable matrices, it is allowed to print any of them.

Examples

input

```
2 3
2 9
5 3 13
```

output

```
YES
3 4 5
6 7 8
```

input

```
3 3
1 7 6
2 15 12
```

output

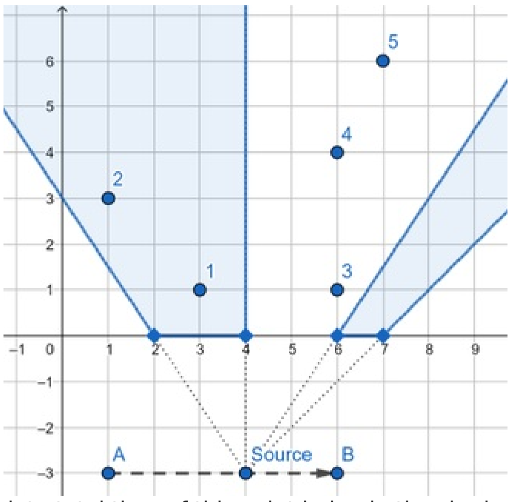
```
NO
```

E. Rest In The Shades

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There is a light source on the plane. This source is so small that it can be represented as point. The light source is moving from point (a, s_y) to the (b, s_y) $(s_y < 0)$ with speed equal to 1 unit per second. The trajectory of this light source is a straight segment connecting these two points.

There is also a fence on OX axis represented as n segments (l_i, r_i) (so the actual coordinates of endpoints of each segment are $(l_i, 0)$ and $(r_i, 0)$). The point (x, y) is *in the shade* if segment connecting (x, y) and the current position of the light source intersects or touches with any segment of the fence.



You are given q points. For each point calculate total time of this point being in the shade, while the light source is moving from (a, s_y) to the (b, s_y).

Input

First line contains three space separated integers s_y, a and b $(-10^9 \le s_y < 0, 1 \le a < b \le 10^9)$ — corresponding coordinates of the light source.

Second line contains single integer n $(1 \le n \le 2 \cdot 10^5)$ — number of segments in the fence.

Next n lines contain two integers per line: l_i and r_i $(1 \le l_i < r_i \le 10^9, r_{i-1} < l_i)$ — segments in the fence in increasing order. Segments don't intersect or touch each other.

Next line contains single integer q $(1 \le q \le 2 \cdot 10^5)$ — number of points to check.

Next q lines contain two integers per line: x_i and y_i $(1 \le x_i, y_i \le 10^9)$ — points to process.

Output

Print q lines. The i-th line should contain one real number — total time of the i-th point being in the shade, while the light source is moving from (a, s_y) to the (b, s_y). The answer is considered as correct if its absolute of relative error doesn't exceed 10^{-6}.

Example

input

```
-3 1 6
2
2 4
6 7
5
3 1
1 3
6 1
6 4
7 6
```

output

```
5.000000000000000
3.000000000000000
0.000000000000000
1.500000000000000
2.000000000000000
```

Note

- The 1-st point is always in the shade;
- the 2-nd point is in the shade while light source is moving from $(3, -3)$ to $(6, -3)$;
- the 3-rd point is in the shade while light source is at point $(6, -3)$.
- the 4-th point is in the shade while light source is moving from $(1, -3)$ to $(2.5, -3)$ and at point $(6, -3)$;
- the 5-th point is in the shade while light source is moving from $(1, -3)$ to $(2.5, -3)$ and from $(5.5, -3)$ to $(6, -3)$;

F. Road Projects

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

There are n cities in the country of Berland. Some of them are connected by bidirectional roads in such a way that there exists *exactly one* path, which visits each road no more than once, between every pair of cities. Each road has its own length. Cities are numbered from 1 to n.

The travelling time between some cities v and u is the total length of the roads on the shortest path from v to u.

The two most important cities in Berland are cities 1 and n.

The Berland Ministry of Transport decided to build a single new road to decrease the traffic between the most important cities. However, lots of people are used to the current travelling time between the most important cities, so the new road shouldn't change it too much.

The new road can only be built between such cities v and u that $v \neq u$ and v and u aren't already connected by some road.

They came up with m possible projects. Each project is just the length x of the new road.

Polycarp works as a head analyst at the Berland Ministry of Transport and it's his job to deal with all those m projects. For the i-th project he is required to choose some cities v and u to build the new road of length x_i between such that the travelling time between the most important cities is **maximal possible**.

Unfortunately, Polycarp is not a programmer and no analyst in the world is capable to process all projects using only pen and paper.

Thus, he asks you to help him to calculate the maximal possible travelling time between the most important cities for each project. Note that the choice of v and u can differ for different projects.

Input

The first line contains two integers n and m ($3 \le n \le 3 \cdot 10^5$, $1 \le m \le 3 \cdot 10^5$) — the number of cities and the number of projects, respectively.

Each of the next $n - 1$ lines contains three integers v_i, u_i and w_i ($1 \le v_i, u_i \le n$, $1 \le w_i \le 10^9$) — the description of the i-th road. It is guaranteed that there exists *exactly one* path, which visits each road no more than once, between every pair of cities.

Each of the next m lines contains a single integer x_j ($1 \le x_j \le 10^9$) — the length of the road for the j-th project.

Output

Print m lines, the j-th line should contain a single integer — the maximal possible travelling time between the most important cities for the j-th project.

Example

input

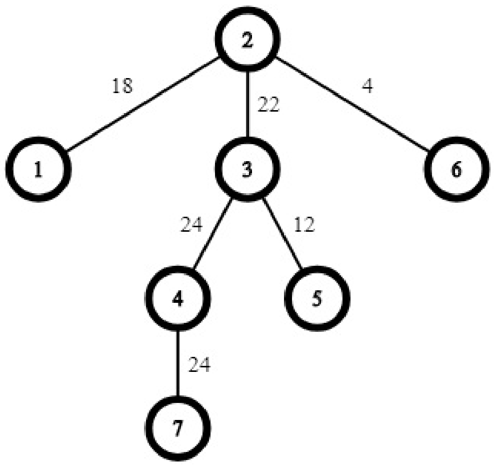
```
7 2
1 2 18
2 3 22
3 4 24
4 7 24
2 6 4
3 5 12
1
100
```

output

```
83
88
```

Note

The road network from the first example:



You can build the road with length 1 between cities 5 and 6 to get 83 as the travelling time between 1 and 7 ($1 \to 2 \to 6 \to 5 \to 3 \to 4 \to 7 = 18 + 4 + 1 + 12 + 24 + 24 = 83$). Other possible pairs of cities will give answers less or equal to 83.

G. Appropriate Team

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Since next season are coming, you'd like to form a team from two or three participants. There are n candidates, the i-th candidate has rank a_i. But you have weird requirements for your teammates: if you have rank v and have chosen the i-th and j-th candidate, then $GCD(v, a_i) = X$ and $LCM(v, a_j) = Y$ must be met.

You are very experienced, so you can change your rank to any non-negative integer but X and Y are tied with your birthdate, so they are fixed.

Now you want to know, how many are there pairs (i, j) such that there exists an integer v meeting the following constraints: $GCD(v, a_i) = X$ and $LCM(v, a_j) = Y$. It's possible that $i = j$ and you form a team of two.

GCD is the greatest common divisor of two number, LCM — the least common multiple.

Input

First line contains three integers n, X and Y ($1 \le n \le 2 \cdot 10^5$, $1 \le X \le Y \le 10^{18}$) — the number of candidates and corresponding constants.

Second line contains n space separated integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^{18}$) — ranks of candidates.

Output

Print the only integer — the number of pairs (i, j) such that there exists an integer v meeting the following constraints: $GCD(v, a_i) = X$ and $LCM(v, a_j) = Y$. It's possible that $i = j$.

Examples

input

```
12 2 2
1 2 3 4 5 6 7 8 9 10 11 12
```

output

```
12
```

input

```
12 1 6
```

```
1 3 5 7 9 11 12 10 8 6 4 2
```

output

```
30
```

Note

In the first example next pairs are valid: $a_j = 1$ and $a_i = [2, 4, 6, 8, 10, 12]$ or $a_j = 2$ and $a_i = [2, 4, 6, 8, 10, 12]$. The v in both cases can be equal to 2.

In the second example next pairs are valid:

- $a_j = 1$ and $a_i = [1, 5, 7, 11]$;
- $a_j = 2$ and $a_i = [1, 5, 7, 11, 10, 8, 4, 2]$;
- $a_j = 3$ and $a_i = [1, 3, 5, 7, 9, 11]$;
- $a_j = 6$ and $a_i = [1, 3, 5, 7, 9, 11, 12, 10, 8, 6, 4, 2]$.

Codeforces (c) Copyright 2010-2022 Mike Mirzayanov
The only programming contests Web 2.0 platform