Bubble Cup 9 - Finals [Online Mirror]

A. Festival Organization

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

The Prodiggers are quite a cool band and for this reason, they have been the surprise guest at the ENTER festival for the past 80 years. At the beginning of their careers, they weren't so successful, so they had to spend time digging channels to earn money; hence the name. Anyway, they like to tour a lot and have surprising amounts of energy to do extremely long tours. However, they hate spending two consecutive days without having a concert, so they would like to avoid it.

A *tour* is defined by a sequence of concerts and days-off. You need to count in how many ways The Prodiggers can select k different tours of the same length between l and r.

For example if $k = 2$, $l = 1$ and $r = 2$, if we define concert day as {1} and day-off as {0}, here are all possible tours: {0}, {1}, {00}, {01}, {10}, {11}. But tour 00 can not be selected because it has 2 days-off in a row. Now, we need to count in how many ways we can select $k = 2$ tours **of the same length** in range $[1;2]$. Here they are: {0,1}; {01,10}; {01,11}; {10,11}.

Since their schedule is quite busy, they want you to tell them in how many ways can do that, modulo $1\ 000\ 000\ 007$ $(10^9 + 7)$.

Input

The first line of the input contains three integers k, l and r $(1 \le k \le 200,\ 1 \le l \le r \le 10^{18})$.

Output

Output a single number: the number of ways to select k different tours of the same length, modulo $1\ 000\ 000\ 007$.

Example

input

```
1 1 2
```

output

```
5
```

B. R3D3's Summer Adventure

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

R3D3 spent some time on an internship in MDCS. After earning enough money, he decided to go on a holiday somewhere far, far away. He enjoyed suntanning, drinking alcohol-free cocktails and going to concerts of popular local bands. While listening to "The White Buttons" and their hit song "Dacan the Baker", he met another robot for whom he was sure is the love of his life. Well, his summer, at least. Anyway, R3D3 was too shy to approach his potential soulmate, so he decided to write her a love letter. However, he stumbled upon a problem. Due to a terrorist threat, the Intergalactic Space Police was monitoring all letters sent in the area. Thus, R3D3 decided to invent his own alphabet, for which he was sure his love would be able to decipher.

There are n letters in R3D3's alphabet, and he wants to represent each letter as a sequence of '`0`' and '`1`', so that no letter's sequence is a prefix of another letter's sequence. Since the Intergalactic Space Communications Service has lately introduced a tax for invented alphabets, R3D3 must pay a certain amount of money for each bit in his alphabet's code (check the sample test for clarifications). He is too lovestruck to think clearly, so he asked you for help.

Given the costs c_0 and c_1 for each '`0`' and '`1`' in R3D3's alphabet, respectively, you should come up with a coding for the alphabet (with properties as above) with minimum total cost.

Input

The first line of input contains three integers n $(2 \le n \le 10^8)$, c_0 and c_1 $(0 \le c_0, c_1 \le 10^8)$ — the number of letters in the alphabet, and costs of '`0`' and '`1`', respectively.

Output

Output a single integer — minimum possible total a cost of the whole alphabet.

Example

input

```
4 1 2
```

output

```
12
```

Note

There are 4 letters in the alphabet. The optimal encoding is "`00`", "`01`", "`10`", "`11`". There are 4 zeroes and 4 ones used, so the total cost is $4 \cdot 1 + 4 \cdot 2 = 12$.

C. Potions Homework

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Harry Water, Ronaldo, Her-my-oh-knee and their friends have started a new school year at their MDCS School of Speechcraft and Misery. At the time, they are very happy to have seen each other after a long time. The sun is shining, birds are singing, flowers are blooming, and their Potions class teacher, professor Snipe is sulky as usual. Due to his angst fueled by disappointment in his own life, he has given them a lot of homework in Potions class.

Each of the n students has been assigned a single task. Some students do certain tasks faster than others. Thus, they want to redistribute the tasks so that each student still does exactly one task, and that all tasks are finished. Each student has their own laziness level, and each task has its own difficulty level. Professor Snipe is trying hard to improve their work ethics, so each student's laziness level is equal to their task's difficulty level. Both sets of values are given by the sequence a, where a_i represents both the laziness level of the i-th student and the difficulty of his task.

The time a student needs to finish a task is equal to the product of their laziness level and the task's difficulty. They are wondering, what is the minimum possible total time they must spend to finish all tasks if they distribute them in the optimal way. Each person should receive one task and each task should be given to one person. Print the answer modulo $10\,007$.

Input

The first line of input contains integer n ($1 \le n \le 100\,000$) — the number of tasks. The next n lines contain exactly one integer number a_i ($1 \le a_i \le 100\,000$) — both the difficulty of the initial task and the laziness of the i-th students.

Output

Print the minimum total time to finish all tasks modulo $10\,007$.

Example

input

```
2
1
3
```

output

```
6
```

Note

In the first sample, if the students switch their tasks, they will be able to finish them in $3 + 3 = 6$ time units.

D. Dexterina’s Lab

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Dexterina and Womandark have been arch-rivals since they’ve known each other. Since both are super-intelligent teenage girls, they’ve always been trying to solve their disputes in a peaceful and nonviolent way. After god knows how many different challenges they’ve given to one another, their score is equal and they’re both desperately trying to best the other in various games of wits. This time, Dexterina challenged Womandark to a game of Nim.

Nim is a two-player game in which players take turns removing objects from distinct heaps. On each turn, a player must remove at least one object, and may remove any number of objects from a single heap. The player who can't make a turn loses. By their agreement, the sizes of piles are selected randomly from the range $[0, x]$. Each pile's size is taken independently from the same probability distribution that is known before the start of the game.

Womandark is coming up with a brand new and evil idea on how to thwart Dexterina’s plans, so she hasn’t got much spare time. She, however, offered you some tips on looking fabulous in exchange for helping her win in Nim. Your task is to tell her what is the probability that the first player to play wins, given the rules as above.

Input

The first line of the input contains two integers n $(1 \leq n \leq 10^9)$ and x $(1 \leq x \leq 100)$ — the number of heaps and the maximum number of objects in a heap, respectively. The second line contains $x + 1$ real numbers, given with up to 6 decimal places each: $P(0), P(1), \dots , P(X)$. Here, $P(i)$ is the probability of a heap having exactly i objects in start of a game. It's guaranteed that the sum of all $P(i)$ is equal to 1.

Output

Output a single real number, the probability that the first player wins. The answer will be judged as correct if it differs from the correct answer by at most 10^{-6}.

Example

input

```
2 2
0.500000 0.250000 0.250000
```

output

```
0.62500000
```

E. Paint it really, really dark gray

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

I see a pink boar and I want it painted black. Black boars look much more awesome and mighty than the pink ones. Since Jaggy became the ruler of the forest, he has been trying his best to improve the diplomatic relations between the forest region and the nearby ones.

Some other rulers, however, have requested too much in return for peace between their two regions, so he realized he has to resort to intimidation. Once a delegate for diplomatic relations of a neighboring region visits Jaggy's forest, if they see a whole bunch of black boars, they might suddenly change their mind about attacking Jaggy. Black boars are really scary, after all.

Jaggy's forest can be represented as a tree (connected graph without cycles) with n vertices. Each vertex represents a boar and is colored either black or pink. Jaggy has sent a squirrel to travel through the forest and paint all the boars black. The squirrel, however, is quite unusually trained and while it traverses the graph, it changes the color of every vertex it visits, regardless of its initial color: pink vertices become black and black vertices become pink.

Since Jaggy is too busy to plan the squirrel's route, he needs your help. He wants you to construct a walk through the tree starting from vertex 1 such that in the end all vertices are black. A walk is a sequence of vertices, such that every consecutive pair has an edge between them in a tree.

Input

The first line of input contains integer n ($2 \le n \le 200\,000$), denoting the number of vertices in the tree. The following n lines contains n integers, which represent the color of the nodes.

If the i-th integer is 1, if the i-th vertex is black and -1 if the i-th vertex is pink.

Each of the next $n - 1$ lines contains two integers, which represent the indexes of the vertices which are connected by the edge. Vertices are numbered starting with 1.

Output

Output path of a squirrel: output a sequence of visited nodes' indexes in order of visiting. In case of all the nodes are initially black, you should print 1. Solution is guaranteed to exist. If there are multiple solutions to the problem you can output any of them provided length of sequence is not longer than 10^7.

Example

input

```
5
1
1
-1
1
-1
2 5
4 3
2 4
4 1
```

output

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

Note

At the beginning squirrel is at node 1 and its color is black. Next steps are as follows:

- From node 1 we walk to node 4 and change its color to pink.
- From node 4 we walk to node 2 and change its color to pink.
- From node 2 we walk to node 5 and change its color to black.
- From node 5 we return to node 2 and change its color to black.
- From node 2 we walk to node 4 and change its color to black.
- We visit node 3 and change its color to black.
- We visit node 4 and change its color to pink.
- We visit node 1 and change its color to pink.
- We visit node 4 and change its color to black.
- We visit node 1 and change its color to black.

F. Heroes of Making Magic III

time limit per test: 3 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

I'm strolling on sunshine, yeah-ah! And doesn't it feel good! Well, it certainly feels good for our Heroes of Making Magic, who are casually walking on a one-directional road, fighting imps. Imps are weak and feeble creatures and they are not good at much. However, Heroes enjoy fighting them. For fun, if nothing else.

Our Hero, Ignatius, simply adores imps. He is observing a line of imps, represented as a zero-indexed array of integers a of length n, where a_i denotes the number of imps at the i-th position. Sometimes, imps can appear out of nowhere. When heroes fight imps, they select a segment of the line, start at one end of the segment, and finish on the other end, without ever exiting the segment. They can move exactly one cell left or right from their current position and when they do so, they defeat one imp on the cell that they moved to, so, the number of imps on that cell decreases by one. This also applies when heroes appear at one end of the segment, at the beginning of their walk.

Their goal is to defeat all imps on the segment, without ever moving to an empty cell in it (without imps), since they would get bored. Since Ignatius loves imps, he doesn't really want to fight them, so no imps are harmed during the events of this task. However, he would like you to tell him whether it would be possible for him to clear a certain segment of imps in the above mentioned way if he wanted to.

You are given q queries, which have two types:

- $1\ a\ b\ k$ — denotes that k imps appear at each cell from the interval $[a, b]$
- $2\ a\ b$ - asks whether Ignatius could defeat all imps on the interval $[a, b]$ in the way described above

Input

The first line contains a single integer n ($1 \le n \le 200\ 000$), the length of the array a. The following line contains n integers $a_1, a_2, \dots, a_n$ ($0 \le a_i \le 5\ 000$), the initial number of imps in each cell. The third line contains a single integer q ($1 \le q \le 300\ 000$), the number of queries. The remaining q lines contain one query each. Each query is provided by integers a, b and, possibly, k ($0 \le a \le b < n$, $0 \le k \le 5\ 000$).

Output

For each second type of query output 1 if it is possible to clear the segment, and 0 if it is not.

Example

input

```
3
2 2 2
3
2 0 2
1 1 1 1
2 0 2
```

output

```
0
1
```

Note

For the first query, one can easily check that it is indeed impossible to get from the first to the last cell while clearing everything. After we add 1 to the second position, we can clear the segment, for example by moving in the following way: .

G. Underfail

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You have recently fallen through a hole and, after several hours of unconsciousness, have realized you are in an underground city. On one of your regular, daily walks through the unknown, you have encountered two unusually looking skeletons called Sanz and P'pairus, who decided to accompany you and give you some puzzles for seemingly unknown reasons.

One day, Sanz has created a crossword for you. Not any kind of crossword, but a 1D crossword! You are given m words and a string of length n. You are also given an array p, which designates how much each word is worth — the i-th word is worth p_i points. Whenever you find one of the m words in the string, you are given the corresponding number of points. Each position in the crossword can be used at most x times. A certain word can be counted at different places, but you cannot count the same appearance of a word multiple times. If a word is a substring of another word, you can count them both (presuming you haven't used the positions more than x times).

In order to solve the puzzle, you need to tell Sanz what's the maximum achievable number of points in the crossword. There is no need to cover all postions, just get the maximal score! Crossword and words contain only lowercase English letters.

Input

The first line of the input contains a single integer n ($1 \le n \le 500$) — the length of the crossword. The second line contains the crossword string. The third line contains a single integer m ($1 \le m \le 100$) — the number of given words, and next m lines contain description of words: each line will have a string representing a non-empty word (its length doesn't exceed the length of the crossword) and integer p_i ($0 \le p_i \le 100$). Last line of the input will contain x ($1 \le x \le 100$) — maximum number of times a position in crossword can be used.

Output

Output single integer — maximum number of points you can get.

Example

input

```
6
abacba
2
aba 6
ba 3
3
```

output

```
12
```

Note

For example, with the string "abacba", words "aba" (6 points) and "ba" (3 points), and $x = 3$, you can get at most 12 points - the word "aba" appears once ("abacba"), while "ba" appears two times ("abacba"). Note that for $x = 1$, you could get at most 9 points, since you wouldn't be able to count both "aba" and the first appearance of "ba".

H. Pokermon League challenge

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Welcome to the world of Pokermon, yellow little mouse-like creatures, who absolutely love playing poker!

Yeah, right…

In the ensuing Pokermon League, there are n registered Pokermon trainers, and t existing trainer teams each of which belongs to one of two conferences. Since there is a lot of jealousy between trainers, there are e pairs of trainers who hate each other. Their hate is mutual, there are no identical pairs among these, and no trainer hates himself (the world of Pokermon is a joyful place!). Each trainer has a wish-list of length l_i of teams he'd like to join.

Your task is to divide players into teams and the teams into two conferences, so that:

- each trainer belongs to exactly one team;
- no team is in both conferences;
- total hate between conferences is at least $e / 2$;
- every trainer is in a team from his wish-list.

Total hate between conferences is calculated as the number of pairs of trainers from teams from different conferences who hate each other.

Input

The first line of the input contains two integer n $(4 \le n \le 50\,000)$ and e $(2 \le e \le 100\,000)$ — the total number of Pokermon trainers and the number of pairs of trainers who hate each other.

Pokermon trainers are numbered from 1 to n. Next e lines contain two integers a and b $(1 \le a, b \le n)$ indicating that Pokermon trainers a and b hate each other. Next $2n$ lines are in a following format. Starting with Pokermon trainer 1, for each trainer in consecutive order: first number l_i $(16 \le l_i \le 20)$ — a size of Pokermon trainers wish list, then l_i positive integers $t_{i,j}$ $(1 \le t_{i,j} \le T)$, providing the teams the i-th trainer would like to be on.

Each trainers wish list will contain each team no more than once. Teams on the wish lists are numbered in such a way that the set of all teams that appear on at least 1 wish list is set of consecutive positive integers $\{1, 2, 3, \ldots, T\}$. Here T might be up to $1\,000\,000$.

Output

Print two lines. The first line should contain n numbers, specifying for each trainer the team he is in.

The second line should contain T numbers, specifying the conference for each team (1 or 2).

Example

input

```
4 3
1 2
2 3
4 1
16
1 2 3 4 5 6 7 8 9 10 11 12 13 14 16 15
16
2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18
16
2 3 4 5 6 7 8 9 10 11 12 13 14 15 18 19
16
1 2 3 4 5 6 7 8 9 10 11 12 13 14 16 19
```

output

```
16 15 19 14
2 2 2 1 1 1 2 1 1 2 1 1 1 2 2 1 1 1 1
```

Note

I. Cowboy Beblop at his computer

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Cowboy Beblop is a funny little boy who likes sitting at his computer. He somehow obtained two elastic hoops in the shape of 2D polygons, which are not necessarily convex. Since there's no gravity on his spaceship, the hoops are standing still in the air. Since the hoops are very elastic, Cowboy Beblop can stretch, rotate, translate or shorten their edges as much as he wants.

For both hoops, you are given the number of their vertices, as well as the position of each vertex, defined by the X , Y and Z coordinates. The vertices are given in the order they're connected: the 1st vertex is connected to the 2nd, which is connected to the 3rd, etc., and the last vertex is connected to the first one. Two hoops are connected if it's impossible to pull them to infinity in different directions by manipulating their edges, without having their edges or vertices intersect at any point – **just like when two links of a chain are connected**. **The polygons' edges do not intersect or overlap**.

To make things easier, we say that two polygons are **well-connected**, if the edges of one polygon cross the area of the other polygon in two different directions (from the upper and lower sides of the plane defined by that polygon) a different number of times.

Cowboy Beblop is fascinated with the hoops he has obtained and he would like to know whether they are well-connected or not. Since he's busy playing with his dog, Zwei, he'd like you to figure it out for him. He promised you some sweets if you help him!

Input

The first line of input contains an integer n $(3 \le n \le 100\ 000)$, which denotes the number of edges of the first polygon. The next N lines each contain the integers x, y and z ($-1\ 000\ 000 \le x, y, z \le 1\ 000\ 000$) — coordinates of the vertices, in the manner mentioned above. The next line contains an integer m $(3 \le m \le 100\ 000)$, denoting the number of edges of the second polygon, followed by m lines containing the coordinates of the second polygon's vertices.

It is guaranteed that both polygons are simple (no self-intersections), and in general that **the obtained polygonal lines do not intersect each other**. Also, you can assume that no 3 consecutive points of a polygon lie on the same line.

Output

Your output should contain only one line, with the words "YES" or "NO", depending on whether the two given polygons are well-connected.

Example

input

```
4
0 0 0
2 0 0
2 2 0
0 2 0
4
1 1 -1
1 1 1
1 3 1
1 3 -1
```

output

```
YES
```

Note

On the picture below, the two polygons are well-connected, as the edges of the vertical polygon cross the area of the horizontal one exactly once in one direction (for example, from above to below), and zero times in the other (in this case, from below to above). Note that the polygons do not have to be parallel to any of the xy-,xz-,yz- planes in general.