Problem A. The One Polynomial Man

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	256 mebibytes

Is it a *programming* contest?

You are given a prime number p and two subsets S and V of residues from 0 to $p-1$.

Your task is to find the number of pairs (a, b) that satisfy the following set of equations:

- $\left(\prod_{z \in V} \left(\frac{(2a+3b)^2+5a^2}{(3a+b)^2} + \frac{(2a+5b)^2+3b^2}{(3a+2b)^2} - z\right)\right) \equiv 0$
- $a \in S$
- $b \in S$

All operations are performed modulo p. Note that, when $a \neq b$, the pairs (a, b) and (b, a) are considered different.

Division by zero is not allowed: when any of the two denominators turns into a zero, the congruence is considered false.

Input

The first line contains a single integer p ($2 \le p \le 10^6$, p is prime).

The second line contains a single integer n: the size of S ($0 \le n \le p$).

The third line contains n distinct integers $S_1, S_2, \ldots, S_n$: the elements of S ($0 \le S_i \le p-1$).

The fourth line contains a single integer m: the size of V ($0 \le m \le p$).

The fifth line contains m distinct integers $V_1, V_2, \ldots, V_m$: the elements of V ($0 \le V_i \le p-1$).

Output

Print one integer: the number of solutions.

Examples

standard input	standard output
7 4 0 4 5 6 2 2 3	8
19 10 0 3 4 5 8 9 13 14 15 18 10 2 3 5 9 10 11 12 13 14 15	42

Problem B. Alexey the Sage of The Six Paths

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

The Party of Cyans and their honourable king Alexey decided to challenge Gennady in an upcoming CodeForces round. There are n party members in the first group, numbered from 1 to n, and n party members in the second group, numbered from $n+1$ to $2n$. There will be m problems in the round.

Alexey will distribute problems in the following way: for each problem, Alexey will assign exactly one party member from the first group and exactly one party member from the second group to work on this problem. After that, each party member will select exactly one problem to solve during the competition among the ones he was assigned to. A problem will be solved only if two party members will select it.

The party members will try their best and always maximize the number of solved problems.

However, members of the Party of Cyans don't solve problems for free. Let c be the number of problems assigned to party member i. Then the amount of money Alexey will have to pay to that party member is $p_{i,c}$.

Due to the competition rules, Alexey will beat Gennady only if his party solves at least l and at most r problems.

Help Alexey to win this competition while paying the minimum possible amount of money to party members.

Input

The first line contains integers n, m, l, and r: the number of party members in one group, the number of problems, and the left and right bounds ($1 \le n \le 30$, $0 \le m \le 30$, $0 \le l \le r \le n$).

The next $2 \cdot n$ lines contain salary requirements for each party member: i-th of them contains $m+1$ integers $p_{i,0}, p_{i,1}, \ldots, p_{i,m}$ ($-10^9 \le p_{i,j} \le 10^9$).

Output

If no money can help to beat Gennady, print "`DEFEAT`" on a single line.

Otherwise, on the first line, print one integer: the minimum possible total payment to the party members that will allow Alexey to win the competition. After that, print m more lines. On i-th of them, print two integers a_i and b_i, where a_i is the number of the party member from the first group assigned to the i-th problem, and b_i is the number of the party member from the second group assigned to the i=th problem.

If there are several possible solutions, print any one of them.

Examples

standard input	standard output
2 0 2 2 8 9 3 4	DEFEAT
2 8 2 2 2 5 5 10 -10 -1 3 5 9 8 -10 9 9 0 1 -3 1 -1 0 5 -1 5 3 -9 1 10 6 5 -4 8 -2 2 -8 6 3 -3	-21 1 3 2 4 1 3 1 3 1 3 2 3 2 4 2 4
3 5 2 3 100 75 125 150 175 200 125 100 75 100 125 150 225 200 175 200 225 250 225 200 175 200 225 250 125 100 75 100 125 150 100 75 125 150 175 200	650 1 4 2 5 3 6 2 4 3 5

Problem C. Steel Ball Run

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	256 mebibytes

You are given a tree with n vertices. Each vertex can contain a chip. Initially, all vertices are empty.

You have to handle two types of queries:

1. Place a chip in a vertex
2. Remove the chip from a vertex

After each query, you have to print the *span* of the current configuration of chips.

The span is defined as the minimum number of operations required to move all chips to the same vertex. In one operation, you can move a chip from its vertex to any adjacent vertex. Of course, during this process one vertex can contain multiple chips.

Note that these operations are needed only for the definition of span, and in fact, they are not performed.

Input

The first line contains a single integer n ($1 \le n \le 10^5$), the number of vertices in the tree.

The next $n-1$ lines describe tree edges, one per line. The i-th of them contains two integers u and v ($1 \le u, v \le n$), which are indices of the vertices connected by the i-th edge.

It is guaranteed that these edges form a tree.

The next line contains a single integer q ($1 \le q \le 10^5$), the number of queries.

The next q lines describe queries, one per line. Each query is described as "c v" ($1 \le v \le n$). The character c is equal to "+" for the first type of query and "−" for the second type. The integer v is the index of the vertex to which the query applies. It is guaranteed that, if the query is of the first type, there is no chip in the v-th vertex, and if the query is of the second type, there is a chip in the v-th vertex. It is also guaranteed that after each query there is at least one chip in the tree.

Output

For each query, print a line with a single integer: the span of the tree after applying this query.

Examples

standard input	standard output
3 1 2 2 3 4 + 1 + 3 + 2 - 1	0 2 2 1
6 1 2 2 3 3 4 4 5 2 6 5 + 1 + 4 + 5 - 5 + 6	0 3 4 3 4

Problem D. The Jump from Height of Self-importance to Height of IQ Level

Input file:	*standard input*
Output file:	*standard output*
Time limit:	7 seconds
Memory limit:	512 mebibytes

There are n skyscrapers arranged in a row, the height of the i-th of them is h_i. The numbers h_i form a permutation of integers from 1 to n.

Alexey wants to make a jump using his grappling hook. In order to perform a jump, he needs exactly three skyscrapers: i, j, k, such that $i < j < k$ and $h_i < h_j < h_k$.

In addition, skyscrapers sometimes change their positions. You have to handle q queries:

In the i-th query you are given l_i, r_i, k_i. Skyscraper from every position j such that $l_i \le j \le r_i - k_i$ moves to position $j + k_i$, and skyscraper from every position j such that $r_i - k_i + 1 \le j \le r_i$ moves to position $j + k_i - (r_i - l_i + 1)$. In other words, you need to shift the segment $l_i, \ldots, r_i$ of the skyscrapers cyclically to the right k_i times.

After each query, help Alexey to determine whether he can perform a jump or not.

Input

The first line contains one integer n ($1 \le n \le 120\,000$), the number of skyscrapers.

The second line contains n integers h_i ($1 \le h_i \le n$), the heights of the skyscrapers. The numbers h_i are pairwise distinct.

The third line contains one integer q ($1 \le q \le 120\,000$), the number of queries.

Next q lines contain descriptions of queries: the i-th of them contains three positive integers l_i, r_i, k_i ($1 \le l_i \le r_i \le n$, $0 \le k_i \le r_i - l_i + 1$).

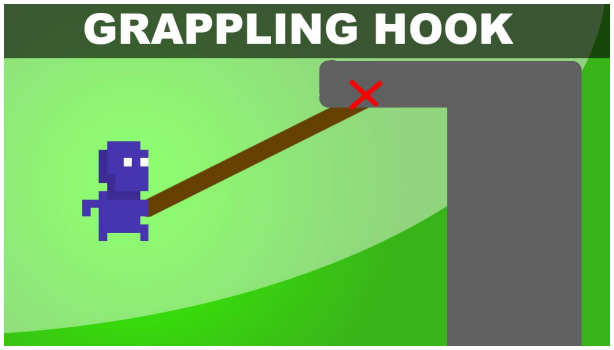
Output

For each query, print a single word on a separate line: "YES" if there are suitable skyscrapers to perform a jump, and "NO" otherwise.

Examples

standard input	standard output
6 2 5 6 1 3 4 1 1 6 5	YES
8 5 1 2 8 7 6 3 4 4 2 4 2 4 5 1 1 3 2 3 8 2	YES YES YES YES
5 4 3 2 5 1 2 3 4 1 1 2 1	NO YES
6 6 5 4 3 2 1 3 1 1 0 1 3 1 2 5 3	NO NO YES

Note



Problem E. Minimums on the Edges

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

You are given an undirected graph with n vertices and m edges. Each vertex can contain several tokens. Initially, there are no tokens in the vertices, but you have s tokens which you can distribute between them.

Let the *capacity* of each edge be the minimal number of tokens in its endpoints. You goal is to maximize the sum of capacities of all edges.

Input

The first line contains three integers n, m and s: the number of vertices, the number of edges and the number of tokens to distribute ($1 \le n \le 18$, $0 \le m \le 100\,000$, $0 \le s \le 100$).

The next m lines describe the edges. The i-th of them describes i-th edge and contains two integers u and v: the indices of vertices it connects ($1 \le u, v \le n$).

It is guaranteed that there are no self-loops in the graph. **However, there can be multiple edges between the same pair of vertices**.

Output

Print n numbers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le s$), where a_i is the number of tokens you put on the i-th vertex. The sum of printed integers must be equal to s. The sum of capacities of all edges must be the maximum possible.

If there are multiple optimal answers, you can print any one of them.

Examples

standard input	standard output
4 4 6 1 2 2 3 3 1 1 4	2 2 2 0
3 7 7 1 2 1 2 1 2 1 3 1 3 2 3 2 3	3 2 2

Note

In the first sample, the sum of capacities is equal to
$\min(2, 2) + \min(2, 2) + \min(2, 2) + \min(2, 0) = 2 + 2 + 2 + 0 = 6$.

In the second sample, the sum of capacities is equal to
$\min(3, 2) + \min(3, 2) + \min(3, 2) + \min(3, 2) + \min(3, 2) + \min(2, 2) + \min(2, 2) = 2 + 2 + 2 + 2 + 2 + 2 + 2 = 14$.

Problem F. IQ Test

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

You are given a set S of integers. Initially, S contains 0, 1, and 2.

You can perform zero or more steps. On each step, you choose two elements (possibly equal) x and y such that $x \in S$ and $y \in S$, and insert the number $x^2 - y$ into the set S.

You can not perform more than 43 steps.

Your task is to get the integer n in your set.

Input

The first line contains a single integer n ($0 \le n \le 10^{18}$), the number you have to get in the set.

Output

For each step, print x and y on a separate line. The condition $0 \le x^2 - y \le 10^{18}$ must be satisfied.

The number of steps must be at most 43. Note that you don't have to minimize it. If there are several possible solutions, print any one of them.

Examples

standard input	standard output
5	1 1 2 1 2 0 3 4
7	1 1 2 1 3 2

Problem G. AtCoder Quality Problem

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

You have a set S of n elements. You want to paint each subset of S either red or blue. For each subset s of S, you know that the cost to paint it red is R_s, and the cost to paint it blue is B_s.

Note: you want to paint subsets, not the elements.

There is only one requirement:

- If a and b are two subsets of S of the same color, the subset $a \cup b$ has the same color as a and b.

Find the minimum total cost to paint all 2^n subsets.

Input

The first line contains a single integer n ($0 \le n \le 20$), the number of elements.

The second line contains 2^n integers $R_0, R_1, \ldots, R_{2^n-1}$ ($-10^9 \le R_i \le 10^9$), the costs to paint subsets red.

The third line contains 2^n integers $B_0, B_1, \ldots, B_{2^n-1}$ ($-10^9 \le B_i \le 10^9$), the costs to paint subsets blue.

The subset i ($0 \le i < 2^n$) is a subset consisting of elements j such that the j-th bit in the binary representation of i is 1.

Output

Print one integer: the minimum cost to paint all subsets.

Examples

standard input	standard output
2 -5 9 9 -5 10 -8 -6 3	-16
3 -15 19 19 -5 30 -3 -16 13 29 -6 -14 -7 24 -5 18 11	-22
0 -129363358 227605714	-129363358
1 -120923470 -355154745 -18478014 104068715	-476078215
3 41 38 35 12 5 15 42 18 37 35 39 13 10 14 11 19	173

Problem H. Mex on DAG

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	256 mebibytes

You are given a directed acyclic graph consisting of n vertices and $2n$ edges. Each edge contains a single integer: more precisely, i-th edge contains the integer $\lfloor \frac{i}{2} \rfloor$. Edges are numbered from 0 to $2n-1$. You need to find a simple path in this graph such that the value of the *mex* function of edges along this path is maximum possible.

We define the value of *mex* of a set of non-negative integers as the smallest non-negative integer which doesn't belong to this set. For example: $mex(0, 1, 3) = 2$.

Input

The first line contains a single integer n ($2 \le n \le 2000$), the number of vertices.

The next $2n$ lines contain description of the edges, from edge number 0 to edge number $2n-1$. The line corresponding to the i-th edge specifies its endpoints: two integers a_i and b_i ($1 \le a_i < b_i \le n$). Recall that the i-th edge contains the integer $\lfloor \frac{i}{2} \rfloor$.

Output

Print a single integer: the largest value of the *mex* function along some simple path in this graph.

Examples

standard input	standard output
8 3 6 2 7 1 3 2 3 6 7 7 8 7 8 4 6 2 7 1 5 2 5 2 8 6 8 7 8 3 5 7 8	4
15 7 15 10 12 13 14 6 8 14 15 9 10 6 13 1 8 6 8 8 9 14 15 13 14 9 13 7 13 14 15 12 14 6 7 3 14 11 14 3 10 10 12 3 8 8 14 13 14 9 11 10 13 6 10 5 10 1 11 13 14	3

Problem I. Find the Vertex

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

You are given a connected undirected graph with n vertices and m edges. The vertices are numbered from 1 to n. The vertex number s is the *initial* vertex. You don't know the number s, but you know all distances from vertex s to every other vertex including itself, taken modulo 3. You have to find the number s.

The distance between two vertices is the length of the shortest path between them. The length of a path is the number of edges in it.

Input

The first line contains two integers n and m ($1 \le n, m \le 500\,000$), the number of vertices and the number of edges.

The second line contains n integers $d_1, d_2, \ldots, d_n$ ($0 \le d_i \le 2$). Here, d_i is the distance between vertices s and i, taken modulo 3.

The next m lines describe the edges. The i-th of these lines describes i-th edge and contains two integers u and v ($1 \le u, v \le n$), the indices of vertices connected by this edge.

It is guaranteed that there are no self-loops and no multiple edges in the graph. Also, it is guaranteed that the graph is connected.

Output

Print the number s: the index of the initial vertex. If there are multiple answers, print any one of them.

Examples

standard input	standard output
5 6 1 0 1 1 2 5 4 1 2 3 2 3 4 4 2 1 5	2
6 6 0 1 2 0 2 1 1 2 2 3 3 4 4 5 5 6 6 1	1

Note

In the first sample, the array of lengths of paths between vertex 2 and all vertices is $[1, 0, 1, 1, 2]$. It is equal to the given array d.

In the second sample, the array of lengths of paths from vertex 1 is $[0, 1, 2, 3, 2, 1]$. If we take each element modulo 3, we will get the array d.

Problem J. Yet Another Mex Problem

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

You are given an array a of length n and an integer k. You need to find the optimal way to divide the given array into several continuous subarrays with lengths no more than k, to maximize your profit. The profit for one subarray is the sum of its elements multiplied by the *mex* value of this subarray. You total profit is the sum of profits for all subarrays.

We define the value of *mex* of a set of non-negative integers as the smallest non-negative integer which doesn't belong to this set. For example: $mex(0, 1, 3) = 2$.

Input

The first line contains two integers: n ($2 \le n \le 200\,000$), the length of the array, and k ($1 \le k \le n$), upper bound for the subarray length.

The second line contains n integers, the elements of the array: the i-th integer is a_i, $0 \le a_i \le n$.

Output

Print a single non-negative integer: the maximum possible profit that can be achieved by dividing the given array into subarrays with lengths no more than k.

Examples

standard input	standard output
5 3 3 4 0 0 3	10
8 4 0 1 2 0 3 1 4 1	26
10 5 0 2 0 1 2 1 0 2 2 1	33