Problem A. Aho-Corasick Automaton

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

Bobo has a tree T with $(n+1)$ nodes labeled with $0, 1, \ldots, n$ rooted at node 0. Each edge is associated with a character.

Let s_i be the concatenation of characters from root to node i. For every i, bobo would like to find f_i such that s_{f_i} is the longest **proper suffix** of s_i.

Note that $s_0 = \varepsilon$ (empty string). String u is a **proper suffix** of v if and only if there exists a non-empty string w such that $wu = v$.

Input

The first line contains one integer n ($1 \le n \le 2 \cdot 10^5$).

The second line contains n integers $p_1, p_2, \ldots, p_n$ where p_i denotes the parent of node i ($0 \le p_i < i$).

The third line contains n integers $c_1, c_2, \ldots, c_n$ where c_i indicates that the edge from node p_i to node i is associated with the c_i-th character from the alphabet ($1 \le c_i \le n$).

It is guaranteed that $(p_i, c_i) \ne (p_j, c_j)$ for all $i \ne j$.

Output

On the first line, print n integers $f_1, f_2, \ldots, f_n$.

Examples

standard input	standard output
2 0 0 1 2	0 0
2 0 1 1 1	0 1

Problem B. All Pair Shortest Path

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	64 mebibytes

Bobo has a directed graph G with n vertices conveniently labeled by $1, 2, \ldots, n$. Let $\delta(i, j)$ be the number of edges on the shortest path from vertex i to vertex j. If the shortest path does not exist, let $\delta(i, j) = n$.

Bobo would like to find $\sum_{i=1}^{n} \sum_{j=1}^{n} \delta^2(i, j)$.

Input

The first line contains an integer n ($1 \le n \le 2000$).

The i-th of the following n lines contains n integers $g_{i,1}, g_{i,2}, \ldots, g_{i,n}$ ($0 \le g_{i,j} \le 1$). If there is an edge from vertex i to vertex j, then $g_{i,j} = 1$. Otherwise, $g_{i,j} = 0$.

Output

Print the integer $\sum_{i=1}^{n} \sum_{j=1}^{n} \delta^2(i, j)$.

Examples

standard input	standard output
3 010 001 100	15
2 10 01	8

Problem C. Chessboard

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	64 mebibytes

Bobo has a chessboard with n rows and m columns. Rows are numbered by $1, 2, \ldots, n$ from top to bottom, and columns are numbered by $1, 2, \ldots, m$ from left to right. Initially, each cell is either black or white.

Bobo performs q operations. The i-th operation changes the color (from black to white and vice versa) of the cell in the intersection of the x_i-th row and y_i-th column. He would like to know the number of connected components after each operation.

Note that cells s and t are in the same connected component if there exist cells $c_0 = s, c_1, \ldots, c_k = t$ for some k where cells c_{i-1} and c_i $(1 \le i \le k)$ share a common edge and all c_i have the same color.

Input

The first line contains three integers n, m and q $(1 \le n, m \le 200, 1 \le q \le 2 \cdot 10^5)$.

The i-th of the following n lines contains m characters $b_{i,1}, b_{i,2}, \ldots, b_{i,m}$. If $b_{i,j} = 1$, then the initial color of cell (i, j) is black, otherwise it is white.

The i-th of the following q lines contains two integers x'_i and y'_i. The actual operation is $(x_i, y_i) = (x'_i \oplus o, y'_i \oplus o)$ where o is the number of connected components **before** the i-th operation $(1 \le x_i \le n, 1 \le y_i \le m)$.

Note that "$\oplus$" stands for bitwise exclusive-or.

Output

For each operation, print an single integer: the number of connected components after this operation.

Examples

standard input	standard output
2 2 2 01 10 5 5 0 0	2 1
1 1 1 0 0 0	1

Problem D. Around the World

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	64 mebibytes

Bobo lives in a world consisting of n cities conveniently labeled by $1, 2, \ldots, n$. Cities are connected by bidirectional roads.

Bobo would like to find a plan $(v_1, v_2, \ldots, v_n)$ where $v_1, v_2, \ldots, v_n$ are n different cities. He could start at city v_1 on day 1, travel to city v_i on day i $(2 \leq i \leq n)$, and return back to city v_1 on day $(n + 1)$. Bobo is lazy, so he will not like a plan which makes him travel by more than k roads on any single day.

Input

The first line contains two integers n and k $(4 \leq n \leq 500, 3 \leq k \leq n - 1)$.

The i-th of the following n lines contains n integers $g_{i,1}, g_{i,2}, \ldots, g_{i,n}$ $(0 \leq g_{i,j} \leq 1, g_{i,j} = g_{j,i})$. If there is a road between city i and city j, then $g_{i,j} = 1$. Otherwise, $g_{i,j} = 0$.

It is guaranteed that each city is reachable from any other city.

Output

On the first line, print n integers $v_1, v_2, \ldots, v_n$ denoting the plan. If there are several possible plans bobo likes, any of them will be accepted.

Example

standard input	standard output
4 3 0100 1010 0101 0010	1 3 2 4

Problem E. Intersection

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	64 mebibytes

Bobo has n lines on a two-dimensional plane. Each pair of them has **exactly one** intersection.

Bobo chose m of the $\binom{n}{2}$ intersections. Now, he would like to find the perimeter of the convex hull of all unchosen intersections.

Recall that the convex hull H of a set of points P is the minimum convex set containing P.

Input

The first line contains two integers n and m ($1 \le n \le 2 \cdot 10^5$, $0 \le m \le 50$).

The i-th of the following n lines contains three integers a_i, b_i and c_i which denote the line $a_i x + b_i y = c_i$ ($|a_i|, |b_i|, |c_i| \le 10^4$, $a_i^2 + b_i^2 > 0$).

The i-th of the following m lines contains two integers x_i and y_i which denote that the intersection of x_i-th and y_i-th lines is chosen by bobo ($1 \le x_i, y_i \le n$, $x_i \ne y_i$).

Output

On the first line, print one real number: the perimeter of the convex hull. Any answer with absolute or relative error less than 10^{-6} is considered correct.

Examples

standard input	standard output
3 0 1 0 0 0 1 0 1 1 1	3.4142135624
3 1 1 0 0 0 1 0 1 1 1 1 2	2.8284271247
1 0 1 1 1	0.0000000000
4 2 1 2 0 1 3 0 1 4 0 1 1 1 1 2 1 3	4.5532455610

Problem F. Data Structure You've Never Heard Of

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	64 mebibytes

Bobo has got a sequence $a_1, a_2, \ldots, a_n$ of d-dimensional binary vectors, and he would like to find the number of non-descending subsequences modulo $(10^9 + 7)$.

Formally, a non-descending subsequence of a is a sequence $(i_1, i_2, \ldots, i_k)$ where $i_1 < i_2 < \ldots < i_k$ and $a_{i_1} \le a_{i_2} \le \ldots \le a_{i_k}$. For two d-dimensional binary vectors $u = (u_1, u_2, \ldots, u_d)$ and $v = (v_1, v_2, \ldots, v_d)$, $u \le v$ if and only if $u_i \le v_i$ holds for all $1 \le i \le d$.

Input

The first line contains two integers n and d ($1 \le n \le 2 \cdot 10^5$, $1 \le d \le 16$).

The i-th of the following n lines contains d integers $a_{i,1}, a_{i,2}, \ldots, a_{i,d}$ ($0 \le a_{i,j} \le 1$).

Output

Print a single integer: the number of non-descending subsequences modulo $(10^9 + 7)$.

Examples

standard input	standard output
3 2 00 00 11	7
4 3 110 100 011 101	5

Problem G. Huffman Coding

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	64 mebibytes

Bobo learned Huffman coding, and he tried to add some restrictions.

Bobo has n words to encode. The i-th word has weight w_i. He wants to encode the i-th word into sequence $S_i = (s_{i,1}, s_{i,2}, \ldots, s_{i,l_i})$ where:

1. $1 \le l_i \le m$.
2. Given $r_1, r_2, \ldots, r_m$, the inequalities $1 \le s_{i,j} \le r_j$ hold for all $1 \le j \le l_i$.
3. For all $i \ne j$, S_i is not a prefix of S_j.

Note that sequence $A = (a_1, a_2, \ldots, a_k)$ is a prefix of sequence $B = (b_1, b_2, \ldots, b_l)$ if and only if $k \le l$ and $a_1 = b_1$, $a_2 = b_2$, $\ldots$, $a_k = b_k$.

Bobo would like to find the minimum of $\sum_{1 \le i \le n} w_i \cdot l_i$.

Input

The first line contains two integers n and m ($1 \le n, m \le 500$).

The second line contains n integers $w_1, w_2, \ldots, w_n$ ($1 \le w_i \le 500$).

The third line contains m integers $r_1, r_2, \ldots, r_m$ ($1 \le r_i \le 500$).

It is guaranteed that $r_1 \cdot r_2 \cdots r_m \ge n$.

Output

Print a single integer: the minimum of $\sum_{1 \le i \le n} w_i \cdot l_i$.

Examples

standard input	standard output
2 2 4 3 2 1	7
3 2 1 2 4 2 2	10

Problem H. Non-Descending Sequence

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	64 mebibytes

Bobo is good at solving Longest Non-Descending Sequence Problem. So he wants to find more.

Given $(a_1, a_2, \ldots, a_n)$, bobo would like to find the number of non-descending sequences $(x_1, x_2, \ldots, x_n)$ (that is, $x_1 \le x_2 \le \ldots \le x_n$) such that $0 \le x_i \le a_i$. As this number can be large, find it modulo 2017.

Input

The first line contains an integer n ($1 \le n \le 2000$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$).

Output

Print one integer: the number of non-descending sequences modulo 2017.

Examples

standard input	standard output
2 1 1	3
3 1 2 4	19

Problem I. Perfect Matching

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

Given an undirected graph $G = (V, E)$ with n vertices and m edges, count the number of perfect matchings modulo $(10^9 + 7)$.

A perfect matching is a **permutation** $\phi : V \to V$ where $(v, \phi(v)) \in E$ and $\phi(\phi(v)) = v$.

Input

The first line contains two integers n and m ($1 \le n \le 30$, $0 \le m \le \frac{n \cdot (n-1)}{2}$).

The i-th of the following m lines contains two integers a_i and b_i which denotes an edge between the a_i-th and b_i-th vertices ($1 \le a_i, b_i \le n$).

It is guaranteed that the graph contains no loops and no multiple edges.

Output

Print one integer: the number of perfect matchings modulo $(10^9 + 7)$.

Examples

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

Problem J. 24 Data Structures You've Ever Heard Of

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	64 mebibytes

Bobo has a permutation p of $\{1, 2, 3, 4\}$ and a permutation a of $\{1, 2, \dots, n\}$. He would like to count the number of subsequences of a similar to p.

That is, to count the number of quadruples (t_1, t_2, t_3, t_4) where:

- $1 \le t_1 < t_2 < t_3 < t_4 \le n$,
- $(p_i - p_j) \cdot (a_{t_i} - a_{t_j}) \ge 0$ for all $1 \le i, j \le 4$.

Input

The first line contains an integer n ($1 \le n \le 2000$).

The second line contains four integers p_1, p_2, p_3 and p_4 ($1 \le p_i \le 4$).

The third line contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le n$).

Output

Print one integer: the number of subsequences of a similar to p.

Examples

standard input	standard output
5 1 2 3 4 1 2 3 4 5	5
8 1 3 2 4 1 2 5 6 3 4 7 8	16