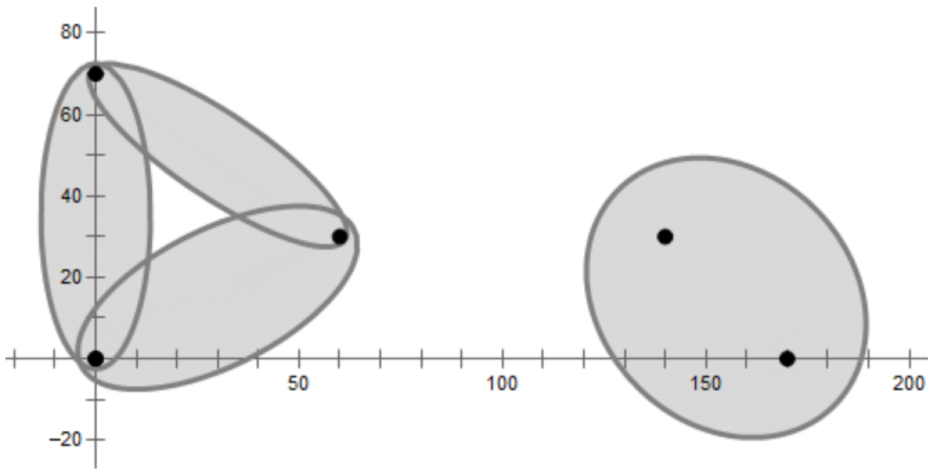
Problem A. Driverless Car II

Input file:	`standard input`
Output file:	`standard output`
Time limit:	10 seconds
Memory limit:	1024 mebibytes

A driverless car company has developed an epoch-making driverless car, and the testers of the company are testing the car on a two-dimension Cartesian plane.

The testers set n distinct transmitters to send signal to the car. When the car moves automatically on the plane, it sometimes break down due to signal attenuation. With great efforts, the testers figure out that the car works well if and only if there exists two different transmitters, the Euclidean distance between one transmitter and the car plus the the Euclidean distance between the other transmitter and the car does not exceed k.



The figure above corresponds to the first sample test case.

As the best programmer in the company, you are now asked to find the total area of positions that allow the car works well.

Input

The first line of the input contains two integers n $(2 \le n \le 2\,000)$ and k $(1 \le k \le 30\,000)$, indicating the number of transmitters and the constraint that allows the driverless car works well.

Each of the following n lines contains two integers x and y $(-10\,000 \le x, y \le 10\,000)$, indicating a transmitter located at coordinate (x, y).

It is guaranteed that no two transmitters coincide, and for any two distinct transmitters, the absolute difference between k and the distance of these two transmitters is at least 0.01.

Output

Output a single real number, indicating the total area of positions that allow the driverless car works well.

Your answer is acceptable if its absolute or relative error does not exceed 10^{-6}. Formally speaking, suppose that your output is a and the jury's answer is b, your output is accepted if and only if $\frac{|a-b|}{\max(1,|b|)} \le 10^{-6}$.

Examples

standard input	standard output

5 75
0 0
170 0
140 30
60 30
0 70
	7881.133252266554

5 40
0 0
170 0
140 30
60 30
0 100
	0.000000000000

5 30000
0 0
1 2
1 5
0 2
0 1
	706960618.428099330689

Problem B. Longest Increasing Subsequence

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

Given an increasing integer sequence $A = a_1, a_2, \cdots, a_n$ of length n whose elements are all distinct, we generate another sequence B with the following algorithm.

Algorithm 1 Sequence Generating Algorithm

```
1: function GENERATE($A$)
2:     $B \leftarrow A$
3:     while true do
4:         $B' \leftarrow B$
5:         Let $S$ be the sequence by sorting $B$ in increasing order
6:         for $i$ in $[1, \text{length of } S)$ do
7:             if $s_i + 1 \neq s_{i+1}$ then                    ▷ $s_i$ is the $i$-th element in $S$
8:                 Add $\lfloor \frac{s_i+s_{i+1}}{2} \rfloor$ to the end of $B'$    ▷ $\lfloor x \rfloor$ is the largest integer not larger than $x$
9:         if $B = B'$ then
10:            break
11:        $B \leftarrow B'$
12:    return $B$
```

It is easy to prove that this algorithm will terminate and that elements of B are all distinct. Calculate the length of the longest increasing subsequence of B.

Input

There is only one test case in each test file.

The first line contains an integer n $(1 \leq n \leq 10^5)$ indicating the length of sequence A.

The second line contains n integers $a_1, a_2, \cdots, a_n$ $(1 \leq a_1 < a_2 < \cdots < a_n \leq 10^{18})$ indicating the given sequence.

Output

Output one line containing one integer indicating the length of the longest increasing subsequence of B.

Example

standard input	standard output
3 1 5 20	11

Note

For the sample test case, $B = \{1, 5, 20, 3, 12, 2, 4, 8, 16, 6, 10, 14, 18, 7, 9, 11, 13, 15, 17, 19\}$. Its longest increasing subsequence is $\{1, 3, 4, 6, 7, 9, 11, 13, 15, 17, 19\}$ of length 11.

Problem C. New Equipments III

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	1024 mebibytes

Little Q's factory recently purchased n pieces of new equipment, labeled by $1, 2, \dots, n$.

There are n workers in the factory, labeled by $1, 2, \dots, n$. Each worker can be assigned to no more than one piece of equipment, and no piece of equipment can be assigned to multiple workers. If Little Q assigns the i-th worker to the j-th piece of equipment, they will bring $p_{i,j}$ profits. However, these workers are not so experienced, so most of the values in matrix p are equal to zero, except m cells. You will be given these m cells.

Now please for every k $(1 \le k \le n)$ find k pairs of workers and pieces of equipment, then assign workers to these pieces of equipment, such that the total profits for these k pairs are maximized.

Input

The input contains only a single case.

The first line contains two integers n and m $(1 \le n \le 50\,000,\ 1 \le m \le 200\,000)$, denoting the number of workers/pieces of new equipment and the number of special cells in p.

Each of the following m lines contains three integers u_i, v_i and w_i $(1 \le u_i, v_i \le n,\ 1 \le w_i \le 5)$, denoting the $p_{u_i,v_i} = w_i$. Each pair of u_i and v_i will be described at most once.

Output

Output n lines, the k-th $(1 \le k \le n)$ of which containing an integer, denoting the maximum possible total profits for k pairs of workers and pieces of equipment.

Examples

standard input	standard output

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

2 3
1 1 5
1 2 2
2 1 2
	5
	5

Problem D. Interesting String Problem

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

JB loves string problems. Here is another interesting string problem created by JB.

Suppose there is a string S, JB will list all the substrings of S. For a substring x, if it occurs multiple times in S, then JB will list it multiple times. After that, JB will sort the strings in the list by lexicographic order, and if two strings are the same, they will be sorted by the positions in S where they occur.After sorting the strings, JB will concatenate the strings by order into one string T.

Now JB will ask you Q queries, each question denoted by one integer k. JB wants you to tell him, for the k^{th} character in string T, where is its position in S?

Input

The first line contains one string $S(1 \le |S| \le 5 \times 10^5)$, contains only lowercase letters.

The second line contains one integer $Q(1 \le Q \le 5 \times 10^5)$.

The following Q lines each contains one integer $k(1 \le k \le |T|)$.

Output

For each query, output one integer denotes the answer.

Examples

standard input	standard output
pbpbppb 3 1 2 3	2 4 7
potatop 3 6 30 60	6 3 4

Problem E. Card Shark

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

There are n decks of cards on the table, the i-th deck consists of a_i cards and the j-th card from top to bottom is $s_{i,j}$. To simplify this problem we only consider cards to be high and low. If the card is a high card then $s_{i,j} = 1$. If it is a low one then $s_{i,j} = 0$.

There are m gamblers sitting around the table waiting for a card game. You are the dealer of the game and your task is to stack all decks of cards into one big pile. You can only control the order between decks. Neither can you insert or remove a card nor can you change the order of cards within a deck.

After the pile is made the cards will be dealt to the gamblers. The i-th gambler will be given the i-th, $(m + i)$-th, $(2m + i)$-th, ..., $(km + i)$-th, ... card from top to bottom. What the other gamblers do not know is that the b-th gambler is actually your boss. To ensure the victory, all cards given to your boss must be high cards and all cards given to the other gamblers must be low cards.

Find an order to stack the decks to fulfill this requirement or report that this is impossible. If a valid answer exists, report the answer with the smallest lexicographical order.

The answer, if it exists, is obviously a permutation of n. We say permutation $P = p_1, p_2, \cdots, p_n$ is lexicographically smaller than permutation $Q = q_1, q_2, \cdots, q_n$ if there exists an integer t such that $p_i = q_i$ for all $1 \le i < t$ and $p_t < q_t$.

Input

There is only one test case in each test file.

The first line of the input contains three integers n, m and b ($1 \le n \le 2 \times 10^5$, $2 \le m \le 2 \times 10^5$, $1 \le b \le m$) indicating the number of decks, the number of gamblers and the boss.

For the following n lines, the i-th line contains a string $s_{i,1}s_{i,2} \cdots s_{i,a_i}$ ($s_{i,j} \in \{0, 1\}$, $1 \le a_i \le 10^6$) indicating the i-th deck.

It's guaranteed that

- There is at least one high card in each deck.
- The total number of cards is divisible by m.
- The total number of cards does not exceed 10^6.

Output

If a valid answer exists, output n integers $d_1, d_2, \cdots, d_n$ separated by a space indicating the answer with the smallest lexicographical order, where d_i is the index of the i-th deck in the pile from top to bottom.

If no vaild answer exists, simply output "-1" (without quotes).

Examples

standard input	standard output

5 4 3
0100010
00100
001000100
0010
0100010
	2 1 3 5 4

4 2 1
010
10101
010
10101
	2 1 4 3

1 5 3 001000010000100	1

2 5 3
01000
00010
	-1

1 5 3 11111	-1

Problem F. Coprime Matrices

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

Given n, m, x, y, w, construct a matrix M satisfying following constraints:

1. the number of rows and columns of M are n, m respectively
2. for each integer $i\,(1 \le i \le nm)$, i appears exactly once in M
3. $M_{x,y} = w$
4. for each entry $M_{i,j}\,(1 < i < n, 1 \le j \le m)$, either $\gcd(M_{i,j}, M_{i-1,j}) = 1$ or $\gcd(M_{i,j}, M_{i+1,j}) = 1$ or both holds
5. for each entry $M_{i,j}\,(1 \le i \le n, 1 < j < m)$, either $\gcd(M_{i,j}, M_{i,j-1}) = 1$ or $\gcd(M_{i,j}, M_{i,j+1}) = 1$ or both holds

If multiple solution exist, print any one of them. If no solution, report it.

Input

Input one line containing five integers $n, m, x, y, w\,(1 \le x \le n \le 300, 1 \le y \le m \le 300, 1 \le w \le nm)$.

Output

If no solution, print "No"(without quotes) in one line.

If solution exists, print "Yes"(without quotes) in the first line. Then print n lines each containing m integers $M_{i,1}, M_{i,2}, \cdots, M_{i,m}$, denoting the answer matrix.

Example

standard input	standard output

`3 3 2 1 3`	`Yes`
	`4 9 2`
	`3 5 7`
	`8 1 6`

Problem G. Factor

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

Let $\mathbb{F}_x$ be the set of all factors of integer x (recall that positive integer f is a factor of x if x is divisible by f). If for all $1 \le y \le x$ there exists a subset of $\mathbb{F}_x$ such that the sum of elements in this subset equals y, then x is considered a good integer.

For example, 6 is good because $\mathbb{F}_6 = \{1, 2, 3, 6\}$ and $4 = 1 + 3$ and $5 = 2 + 3$. 5 is not good because $\mathbb{F}_5 = \{1, 5\}$ and we can't find a subset whose sum equals 2, 3 or 4.

Given an integer n, calculate the number of good integers such that $1 \le x \le n$.

Input

There is only one test case in each test file.

The first and only line contains an integer n ($1 \le n \le 10^{12}$).

Output

Output one line containing one integer indicating the number of good integers such that $1 \le x \le n$.

Examples

standard input	standard output
10	5
20	9
50	17

Problem H. Graph Operation

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

You are given two undirected graphs G and H. Both G and H have exactly n vertices and m edges, and the vertices are labeled from 1 to n. Now, you need to change graph G to graph H. You can perform the following operation any number of times:

- First select four distinct vertices a, b, c, and d. You should ensure that $a \sim b$, $c \sim d$ while $a \nsim c$, $b \nsim d$.
- Delete the edge between a and b, and the one between c and d. Add an edge between a and c and one between b and d.

Here $a \sim b$ means that there exists an edge between a and b, and $a \nsim b$ means that there doesn't exist an edge between a and b.

Note that you can select a different set of a, b, c, d each time. Please determine whether you can change graph G to graph H. If yes you also need to provide the detailed steps.

Input

The first line of the input contains two integers n and m ($4 \le n \le 1000$, $0 \le m \le \binom{n}{2}$) indicating the number of vertices and edges in graph G and H.

For the following m lines, the i-th line contains two integers u and v where $1 \le u \ne v \le n$, indicating that there exists an edge between u and v in graph G.

For the following m lines, the i-th line contains two integers u and v where $1 \le u \ne v \le n$, indicating that there exists an edge between u and v in graph H.

Neither graph G nor H has multi-edges or self-loops.

Output

If you cannot change G to H output "-1" (without quotes).

Otherwise first output an integer r ($0 \le r \le 3 \times 10^6$) in one line indicating the number of operations you need.

For the following r lines, output four integers a_i, b_i, c_i and d_i in the i-th line separated by a space, indicating that for the i-th operation you choose vertices a_i, b_i, c_i and d_i. Note that a_i, b_i, c_i, d_i must be distinct.

Example

standard input	standard output

4 2
1 2
3 4
1 3
2 4
	1
	1 2 3 4

Problem I. Optimal Assortment

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

Potato is a toy retailer. He has n types of toys in his warehouse. Selling the i-th toy can make a profit v_i. He has conducted market research in advance. He knows that a customer buys at most one toy and customer's preference w_i for toy i will be in range $[l_i, r_i]$.

When potato offers a toy set S to the customer, the probability of a customer buying toy $i \in S$ is $\frac{w_i}{w_0+\sum_{i\in S} w_i}$ and the probability of a customer not making a purchase is $\frac{w_0}{w_0+\sum_{i\in S} w_i}$. Specifically, $w_i = 0$ indicates that the customer prefers not to buy such toy. When $w_i = 0$ for $i = 0$ and all $i \in S$, potato gains nothing. Potato wants to choose an optimal set of toys to maximize the mathematic expectation of his profit in the worst case, where w_i can be arbitrarily chosen within the ranges.

Potato is a smart guy and he can easily solve the above problems by himself. He raises a harder question. If there are two kinds of modification operations, the first modification operation will change the range of customer's preference and the second modification operation will change the profit of the toy i. Here one operation will effect all follow-up calculations. Can you quickly answer the the maximum profit after each modification?

Input

The first line contains two integers n and m $(1 \le n, m \le 2 \times 10^5)$ - the number of types of toys and the number of modification operations.

The second line contains n integers $v_1, v_2, \cdots, v_n$ $(1 \le v_i \le 10^6)$ - the profit of each type of toys.

The third line contains $n+1$ integers $l_0, l_1, \cdots, l_n$ - the lower bounds of customer's preference to buy toys.

The fourth line contains $n+1$ integers $r_0, r_1, \cdots, r_n$ $(0 \le l_i \le r_i \le 10^6)$ - the upper bounds of customer's preference to buy toys.

The next m line contains m modification operations, which is in one of the following two types:

- 1 x y z $(0 \le x \le n, 0 \le y \le z \le 10^6)$ - change range of customer's preference to buy toy x to $[y, z]$
- 2 x y $(1 \le x \le n, 1 \le y \le 10^6)$ - change the profit of toy x to y

Output

Print $m + 1$ irreducible fraction (in the form of a/b, where the greatest common divisor of a and b is 1) — the initial profit and the profit after each modification.

Example

standard input	standard output

2 5
4 2
4 3 2
4 3 2
2 1 2
1 1 2 3
1 0 0 0
1 1 0 0
1 2 0 0
	16/9
	10/9
	1/1
	2/1
	2/1
	0/1

Problem J. Cell Tower

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

Cell Tower is an interesting daily puzzle game, `https://www.andrewt.net/puzzles/cell-tower/`.

Here we consider a simplified version. You are given 8×8 square with one character in each cell and a dictionary. Please divide the square into several parts, so that each part is a **connected block** and the characters in this connected block (from top to bottom and from left to right) make up a valid word (i.e., appear in the dictionary).

Two cells A, B are called **connected pair**, if A and B directly share the same side, or there exists another cell C so that both A, C and B, C are connected pairs.

A group of cells is called **connected block** if any pair of cells in this group are **connected pairs** and the size of this group is either 3 or 4.

Input

The first 8 lines of input each contains 8 integers $Sq_{i,j} (0 \le Sq_{i,j} \le 9)$ indicating the given 8×8 square.

In the next line, there is one integer $n (1 \le n \le 11000)$ indicating the size of the dictionary.

In the next n lines, there is a string $S_i (3 \le |S_i| \le 4)$ in each line, describing the word in the dictionary. The character set of S_i is $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Output

Please output the number of valid divisions.

Example

standard input	standard output

1 1 1 1 2 3 3 3
0 4 4 4 2 2 2 3
0 0 5 5 6 6 7 7
0 9 5 5 6 8 7 7
9 9 9 1 6 8 8 8
3 1 1 1 2 2 2 2
4 5 6 0 0 4 4 3
7 8 9 0 0 4 3 3
16
1111
2222
3333
444
0000
5555
6666
7777
8888
9999
111
333
3456
789
3478
569
	2

Problem K. Xiangqi

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

DreamGrid is playing Xiangqi(Chinese Chess) on an infinity 2D plane.

DreamGrid has two black pieces: a horse and a cannon. There is only a red piece - a king - located at the original point $(0, 0)$. DreamGrid wants to eliminate the red king in the minimum steps. To reduce the difficulty, the red king cannot move in the process.

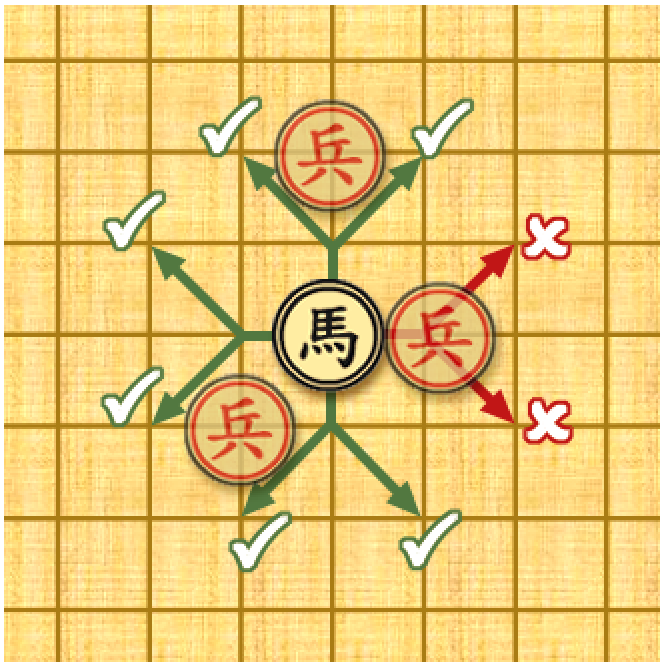
The rules of moving the cannon and the horse is the same as the standard Xiangqi rule, including the 'horse leg' rule. If you don't know that, you can refer to the explanation below.

When you want to move a horse at (x, y), you need to firstly choose two **mutually perpendicular** directions (dx_1, dy_1) and (dx_2, dy_2) which belong to $\{(1, 0), (-1, 0), (0, 1), (0, -1)\}$, and move the horse to $(x + 2dx_1 + dx_2, y + 2dy_1 + dy_2)$. In most cases, the horse can move to eight positions: $(x+2, y+1), (x+2, y-1), (x-2, y+1), (x-2, y-1), (x+1, y+2), (x+1, y-2), (x-1, y+2), (x-1, y-2)$. If there is an opposite piece at the destination of the step, the horse can eliminate it.

The 'horse leg' rule restricts the movement of horses. The horse cannot move to $(x + 2dx_1 + dx_2, y + 2dy_1 + dy_2)$, if there is another piece(either color) at $(x + dx_1, y + dy_1)$.

In a single step, the cannon can either move or attack. It can move any distance horizontally or vertically(select one of them) in a step, but it cannot jump over other pieces when moving. Cannon can only attack other pieces with a 'cannon platform'. Formally, in a direction(up, down, left, right) of the cannon, if there are two or more pieces, and the **second** nearest piece is opposite, the cannon can eliminate the piece and move there.

Attack of cannon



Movement of horse

Can you help DreamGrid eliminate the red king in the minimum steps?

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 500\,000$), indicating the number of test cases. For each test case:

The only line contains four integers x_h, y_h, x_c, y_c ($-10^9 \le x_h, y_h, x_c, y_c \le 10^9$). The initial position of the horse is (x_h, y_h), and the initial position of the cannon is (x_c, y_c).

It's guaranteed that the initial positions of the horse, the cannon, and the king are distinct.

Output

For each test case output one line, indicating the minimum step to eliminate the red king.

Example

standard input	standard output

5
-2 1 5 5
5 0 6 0
2 1 1 1
100 2 -1 0
4 2 1 1
	1
	1
	2
	5
	3