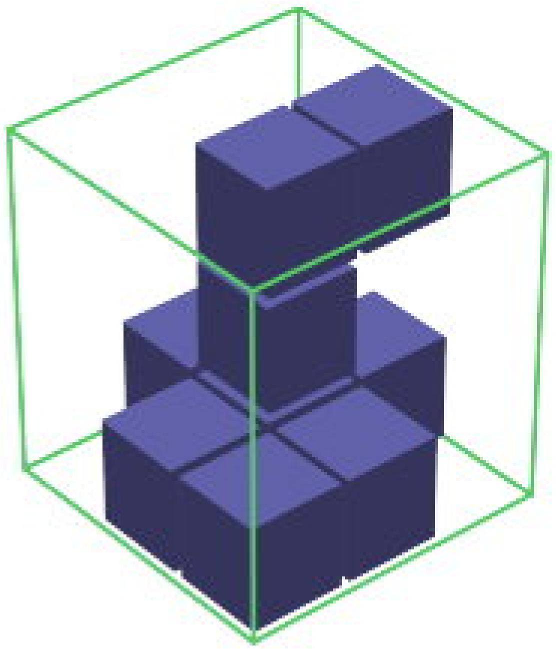
Problem A. Axonometry

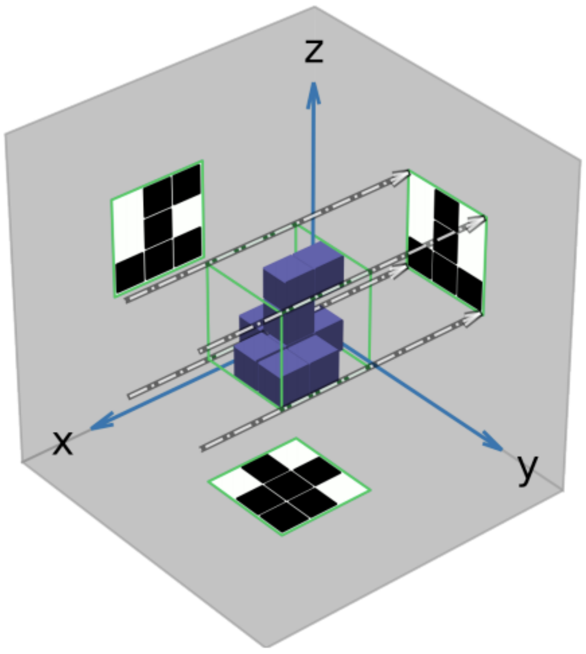
Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

A friend of yours is an artist designing a crystal cubes with its side of integer length. Some of its regions are made opaque through an elaborate laser machining process. Each of the opaque regions is a cube of unit size aligned with integer coordinates.

Figure below depicts an art object as given in the Sample Input 1. Here, the green wires correspond to the edges of the crystal cube. The blue small cubes correspond to the cubic opaque regions inside the crystal.



Art objects are meant to enjoy the silhouettes of its opaque regions projected by three parallel lights perpendicular to its faces. Figure below depicts the cube and its three silhouettes.



Your job is to write a program that decides whether there exists an ICPC that makes the given silhouettes.

Input

The first line of the input contains one integer n — the length of the edge of the cube, an integer between 1 and 100, inclusive.

The cube of size n makes three silhouettes of $n \times n$ squares of dark and bright cells. Cells are dark if they are covered by the shadows of opaque unit cubes, and are bright, otherwise. The $3n$ lines, each with n digits, starting from the second line of the input denote the three silhouettes of the cube, where '`0`' represents a bright cell and '`1`' represents a dark cell. At least one of the digits in the 3n lines is '`1`'.

First comes the data for the silhouette on the yz-plane, where the first line s_1 gives the cells of the silhouettes with the largest z-coordinate value. They are in the order of their y-coordinates. The following lines, s_2, …, s_n, give the cells with the decreasing values of their z-coordinates.

Then comes the data for the silhouette on the zx-plane, where the first line, t_1, gives the cells with the largest x-coordinate value, in the order of their z-coordinates. The following lines, t_2, …, t_n, give the cells with the decreasing values of their x-coordinates.

Finally comes the data for the silhouette on the xy-plane, where the first line, u_1, gives the cells with the largest y-coordinate value, in the order of their x-coordinates. The following lines, u_2, …, u_n, give the cells with the decreasing values of their y-coordinates.

Output

Print "`Yes`" if it is possible to make an ICPC with the given silhouettes. Print "`No`", otherwise.

Examples

standard input	standard output
3 010 010 111 100 111 101 011 111 010	Yes
2 00 01 00 10 10 00	Yes
2 00 00 00 10 10 00	No
2 01 00 00 10 10 00	No

Problem B. Big Treasure

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

You have discovered the document with two integer sequences, the final keys to unlocking the secret of the big treasure! The ink on the document, however, has faded so much that some of the numbers in the sequences are too faint to read. Fortunately, you have heard of a legend on the characteristics of the key sequences that:

1. the numbers in the two sequences are all different and are between 1 and the sum of the lengths of the two sequences, and
2. both sequences are arranged in ascending order.

Please restore the original sequences from the readable numbers and the legend.

Input

The first line consists of two integers n ($1 \le n \le 100$) and m ($1 \le m \le 100$). They are the lengths of the two sequences A and B, respectively. The second line describes the sequence A. Each a_i is either 0, meaning that the i-th element of A is unreadable, or a positive integer $1 \le a_i \le n + m$ meaning that the element is readable and equal to a_i. The third line describes the sequence B in the the same way.

Output

Print a pair of restored sequences A and B in two lines. The sequences should be consistent with the input and the legendary characteristics of the sequences.

The elements of the sequence A should be printed in the first line, separated by a space character. Then the elements of the sequence B should be printed in the second line, in the same manner.

If there are multiple possibilities, print any one of such pairs. At least one sequence pair consistent with the input and conforming to the legend is guaranteed to exist.

Examples

standard input	standard output

3 3
0 0 0
4 0 0
	1 2 3
	4 5 6

6 7
0 5 0 0 0 13
0 0 3 0 8 0 12
	4 5 7 10 11 13
	1 2 3 6 8 9 12

Note

Note that there are other acceptable outputs for the second sample input, such as the one with the positions of 6 and 7 exchanged.

Problem C. Compact Code

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

A maze-like space in underground of the newly opened asteroid was found by a survey a couple of years ago. The project is to investigate the maze more in detail using an exploration robot.

The shape of the maze was fully grasped by a survey with ground penetrating radars. For planning the exploration, the maze is represented as a grid of cells, where the first cell of the first row is the upper left corner of the grid, and the last cell of the last row is the lower right corner.

Each of the grid cell is either vacant, allowing robot's moves to it from an adjacent vacant cell, or filled with rocks, obstructing such moves. The entrance of the maze is located in a cell in the uppermost row and the exit is in a cell in the lowermost row.

The exploration robot is controlled by a program stored in it, which consists of sequentially numbered lines, each containing one of the five kinds of commands described below. The register pc specifies the line number of the program to be executed. Each command specifies an action of the robot and a value to be set to pc.

The robot starts at the entrance of the maze facing downwards, and the value of pc is set to 1. The program commands on the lines specified by pc are executed repetitively, one by one, until the robot reaches the exit of the maze.

When the value of pc exceeds the number of lines of the program by its increment, it is reset to 1. The robot stops on reaching the exit cell, which is the goal of the project.

As the capacity of the program store for the robot is quite limited, the number of lines of the program should be minimal. Your job is to develop a program with the fewest possible number of lines among those which eventually lead the robot to the exit.

Input

The first line contains two integers n ($2 \le n \le 10$) and m ($2 \le m \le 10$). The maze is represented as a grid with n rows and m columns. The next n lines describe the grid. Each of the lines contains a string of length m corresponding to one grid row. The i-th character of the j-th string, either '`.`', '`#`', '`S`' or '`G`', describes the i-th cell of the j-th row. '`.`' means that the cell is vacant and the robot in one of the four adjacent cells can move to it. '`#`' means that the cell is filled obstructing the robot's moves to it. '`S`' means that the cell is the entrance, and '`G`' means that the cell is the exit. These cells are vacant, of course.

It is known that a program exists that leads the robot to the exit.

The list of commands:

- `GOTO l` — set l to pc. The command parameter l is a positive integer less than or equal to the number of lines of the program.
- `IF-OPEN l` — if there is a vacant adjacent cell in its current direction, set l to pc; otherwise, that is, facing a filled cell or a border of the grid, increment pc by one. The command parameter l is a positive integer less than or equal to the number of lines of the program.
- `FORWARD` — if there is a vacant adjacent cell in its current direction, move there; otherwise, stay in the current cell. In either case, increment pc by one.
- `LEFT` — turn 90 degrees to the left without changing the position, and increment pc by one.
- `RIGHT` — turn 90 degrees to the right without changing the position, and increment pc by one.

Output

The first line of the output should have the number of lines of the program. The commands in the program lines should follow, one per each line, in the order of their line numbers. When the command has a parameter, output only one space between the command name and the parameter.

If more than one appropriate program has the fewest lines, whichever is acceptable.

Examples

standard input	standard output
`2 2`	`2`
`.S`	`FORWARD`
`G#`	`LEFT`
`5 2`	`3`
`S.`	`IF-OPEN 3`
`..`	`LEFT`
`..`	`FORWARD`
`..`	
`.G`	
`2 6`	`4`
`..S...`	`RIGHT`
`..#.G#`	`RIGHT`
	`FORWARD`
	`GOTO 2`

Problem D. Drawing Colorful Rectangle

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	512 mebibytes

You are given a set of points on a plane. Each point is colored either red, blue, or green. A rectangle is called colorful if it contains one or more points of every color inside or on its edges. Your task is to find an axis-parallel colorful rectangle with the shortest perimeter. An axis-parallel line segment is considered as a degenerated rectangle and its perimeter is twice the length of the line segment.

Input

The first line contains an integer n ($3 \le n \le 10^5$) representing the number of points on the plane. Each of the following n lines contains three integers x_i, y_i, and c_i satisfying $0 \le x_i \le 10^8$, $0 \le y_i \le 10^8$, and $0 \le c_i \le 2$. Each line represents that there is a point of color c_i (0: red, 1: blue, 2: green) at coordinates (x_i, y_i). It is guaranteed that there is at least one point of every color and no two points have the same coordinates.

Output

Output a single integer in a line which is the shortest perimeter of an axis-parallel colorful rectangle.

Examples

standard input	standard output

4
0 2 0
1 0 0
1 3 1
2 4 2
	8

4
0 0 0
0 1 1
0 2 2
1 2 0
	4

Problem E. Embedding the Polygon

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Given n integers — lengths of the segments. Your task is to embed the polygon with n vertices and with edges equal to those integers into the circle of the minimal radius (i.e. such as all the vertices are placed on the circumference).

Input

First line of the input contains one integer n that indicates the number of edges ($3 \le n \le 1000$). x_k ($k = 1, \ldots, n$) is an integer that indicates the length of the k-th edge ($1 \le x_k \le 6000$).

You may assume the existence of one or more polygons with the specified edge lengths. You can prove that one of such polygons has a circumscribed circle.

Output

Output the minimum radius of a circumscribed circle of a polygon with the specified edge lengths. Absolute/relative error of the output should be within 10^{-7}.

Examples

standard input	standard output
5 3 1 6 1 7	3.54440435
3 500 300 400	250.0
6 2000 3000 4000 2000 3000 4000	3037.33679126
10 602 67 67 67 67 67 67 67 67 67	3003.13981697
3 6000 6000 1	3000.00001042

Problem F. Funny Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	512 mebibytes

Alice and Bob are playing a game of baggage delivery using a toy solar car. A number of poles are placed here and there in the game field.

At the start of a game, the solar car is placed right next to a pole, and the same or another pole is marked as the destination. Bob chooses one of the poles and places the baggage there. Then, Alice plans a route of the car to pick up the baggage and deliver it to the marked destination. Alice chooses the shortest possible route and Bob should choose a pole to maximize the route length.

The solar car can drive arbitrary routes but, as it has no battery cells, it would stop as soon as it gets into shadows. In this game, a point light source is located on the surface of the field, and thus the poles cast infinitely long shadows in the directions opposite to the light source location. The drive route should avoid any of these shadows.

When the initial positions of the solar car and the destination are given, assuming that both players make the best choices, the length of the drive route is uniquely determined.

Let us think of all the possible game configurations with given two sets of poles, one for the start positions of the solar car and the other for the destinations. When both the initial car position and the destination are to be chosen uniformly at random from the corresponding sets, what is the expected route length?

Your task is to compute the expected value of the route length, given the set of the initial positions of the car and that of the destinations.

Input

First line of the input contains one integer n — the number of poles ($3 \le n \le 2000$).

Then n lines follow, each containing two integers x_i and y_i. The poles are numbered 1 through n; the i-th pole is described in i-th line and is placed at integer coordinates (x_i, y_i) ($-1000 \le x_i \le 1000$, $-1000 \le y_i \le 1000$). The point light is at $(0, 0)$. Poles are not placed at $(0, 0)$ nor in a shadow of another pole, and no two poles are placed at the same coordinates.

Then follows the line with one integer p — the number of pairs of sets ($1 \le p \le 10^5$). Then p lines follow; the i-th of them describes i-th pair of sets and is specified by four integers (a_i, b_i, c_i, d_i) ($1 \le a_i \le b_i \le n$, $1 \le c_i \le d_i \le n$). Specifically, the solar car is initially placed at the j-th pole, with j chosen uniformly at random from $\{a_i, \dots b_i\}$, and the destination pole is also chosen uniformly at random from $\{c_i, \dots, d_i\}$.

Output

Output the answer for each pair of sets. Absolute or relative errors less than 10^{-7} are permissible.

Examples

standard input	standard output

```
5
7 0
3 3
0 7
-3 3
-7 0
6
1 1 3 3
3 3 4 4
1 1 5 5
5 5 2 2
2 2 4 4
1 5 1 5
```
	```
	24.000000000000
	20.440306508911
	20.000000000000
	19.000000000000
	15.440306508911
	21.606571644990
	```

Problem G. Graph Modifications

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 mebibytes

An undirected graph is given, each of its nodes associated with a positive integer value. Given a threshold, nodes of the graph are divided into two groups: one consisting of the nodes with values less than or equal to the threshold, and the other consisting of the rest of the nodes. Now, consider a subgraph of the original graph obtained by removing all the edges connecting two nodes belonging to different groups. When both of the node groups are non-empty, the resultant subgraph is disconnected, whether or not the given graph is connected.

Then a number of new edges are added to the subgraph to make it connected, but these edges must connect nodes in different groups, and each node can be incident with at most one new edge. The threshold is called feasible if neither of the groups is empty and the subgraph can be made connected by adding some new edges.

Your task is to find the minimum feasible threshold.

Input

The first line of the input contains two integers n ($2 \le n \le 10^5$) and m ($0 \le m \le \min(10^5, n(n-1)/2)$), the numbers of the nodes and the edges, respectively, of the graph. Nodes are numbered 1 through n. The second line contains n integers l_i ($1 \le l_i \le 10^9$), meaning that the value associated with the node i is l_i. Each of the following m lines contains two integers x_j and y_j ($1 \le x_j < y_j \le n$), meaning that an edge connects the nodes x_j and y_j. At most one edge exists between any two nodes.

Output

Output the minimum feasible threshold value. Output -1 if no threshold values are feasible.

Examples

standard input	standard output
4 2	20
10 20 30 40	
1 2	
3 4	
2 1	3
3 5	
1 2	
3 0	-1
9 2 8	
4 6	-1
5 5 5 5	
1 2	
1 3	
1 4	
2 3	
2 4	
3 4	
7 6	2
3 1 4 1 5 9 2	
2 3	
3 5	
5 6	
1 4	
1 7	
4 7	

Problem H. How to Combine GCD and LCM

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

You are given a sequence of positive integers, followed by a number of instructions specifying updates to be made and queries to be answered on the sequence. Updates and queries are given in an arbitrary order.

Each of the updates replaces a single item in the sequence with a given value. Updates are accumulated: all the following instructions are on the sequence after the specified replacement.

Each of the queries specifies a subsequence of the (possibly updated) sequence and the number of items to exclude from that subsequence. One or more sets of integers will result depending on which of the items are excluded. Each of such sets has the greatest common divisor (GCD) of its members. The answer to the query is the least common multiple (LCM) of the GCDs of all these sets.

Input

The first line has two integers, n and m. n ($1 \le n \le 10^5$) is the length of the integer sequence, and m ($1 \le m \le 10^5$) is the number of instructions. The original integer sequence $a_1, \ldots, a_n$ is given in the second line. $1 \le ai \le 10^6$ holds for $i = 1, \ldots, n$. Each of the following m lines has either an update instruction starting with a letter '`U`', or a query instruction starting with a letter '`Q`'.

An update instruction has the following format: `U` j x. The instruction tells to replace the j-th item of the sequence with an integer x. $1 \le j \le n$ and $1 \le x \le 10^6$ hold. Updates are accumulated: all the instructions below are on the sequence after the updates.

A query instruction has the following format: `Q` l r k. Here, l and r specify the start and the end positions of a subsequence, and k is the number of items to exclude from that subsequence, $b_l, \ldots, b_r$, where $b_1, \ldots, b_n$ is the sequence after applying all the updates that come before the query. ($1 \le l$, $0 \le k \le 2$, and $l + k \le r \le n$ hold.

Output

No output is required for update instructions. For each of the query instructions, output a line with the LCM of the GCDs of the sets of the items in all the subsequences made by excluding k items from the sequence $b_l, \ldots, b_r$.

Examples

standard input	standard output
5 10 12 10 16 28 15 Q 1 3 1 Q 3 4 0 Q 2 2 0 Q 2 5 2 U 3 21 Q 1 3 1 Q 2 4 1 Q 3 5 1 Q 4 4 0 Q 2 5 2	4 4 10 20 6 14 21 28 210

Problem I. International Cookie Packing Contest

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 mebibytes

The problem set document for the International Cookie Packing Contest was kept in a safe at the headquarters of the Gourmendies Foundation. It was considered to be quite secure, but, in the morning of the very day of the contest, the executive director of the foundation found that the document was missing!

The director checked that the document was in the safe when she left the headquarters in the evening of the day before. To open the door of the headquarters office, a valid ID card has to be touched on the reader inside or outside of the door. As the door and its lock are not broken, the thief should have used a valid ID card.

Normally, all the entries and exits through the door are recorded with the ID. The system, however, has been compromised somehow, and some of the recorded ID's are lost.

It is sure that nobody was in the office when the director left, but, many persons visited the office since then to prepare the contest materials. It is sure that the same ID card was used only once for entry and then once for exit.

The director is planning inquiries to grasp all the visits during the night. You are asked to write a program that calculates the number of possible combinations of ID's to fill the lost parts of the records.

Input

The first line contains an integer n ($1 \le n \le 5000$), representing the number of visitors during the night. Each visitor has a unique ID numbered from 1 to n. The following $2n$ lines provide (incomplete) entry and exit records in chronological order. The i-th line ($1 \le i \le 2n$) contains a character c_i and an integer x_i ($0 \le x_i \le n$). Here, c_i denotes the type of the event, where c_i = '`I`' and '`O`' indicate some visitor entered and exited the office, respectively. x_i denotes the visitor ID, where $x_i \ge 1$ indicates the ID of the visitor is x_i, and $x_i = 0$ indicates the ID is lost. At least one of them is 0. It is guaranteed that there is at least one consistent way to fill the lost ID(s) in the records.

Output

Output a single integer in a line which is the number of the consistent ways to fill the lost ID(s) modulo $10^9 + 7$.

Examples

standard input	standard output
4 I 1 I 0 O 0 I 0 O 2 I 4 O 0 O 4	3
3 I 0 I 0 I 0 O 0 O 0 O 0	36

Problem J. Journey Of The Ant

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 mebibytes

A new species of ant, named *Formica sokobanica*, was discovered recently. It attracted entomologists' attention due to its unique habit. These ants do not form colonies. Rather, individuals make up their private nests and keep their food, nuts, in the nests. A nest comprises a lot of small rooms connected with tunnels. They build the rooms only a little bigger than a nut leaving just enough space for air flow; they cannot enter a room with a nut in it. To save the labor, tunnels are so narrow that it exactly fits the size of a nut, and thus nuts should not be left in the tunnels to allow air flow through.

To enter a room with a nut in it, the nut has to be pushed away to any of the vacant rooms adjacent to that room through the tunnel connecting them. When no adjacent rooms are vacant except the room which the ant came from, the nut cannot be pushed away, and thus the ant cannot enter the room.

Dr. Myrmink, one of the entomologists enthused about the ants, has drawn a diagram of a typical nest. The diagram also shows which rooms store nuts in them, and which room the ant is initially in. Your job is to write a program that counts up how many rooms the ant can reach and enter. Pushing a nut into one of the vacant adjacent rooms may make some of the rooms unreachable, while choosing another room to push into may keep the rooms reachable. There can be many combinations of such choices. In such cases, all the rooms should be counted that are possibly reached by one or more choice combinations.

You may assume that there is no nut in the room the ant is initially in, and that there is no cyclic path in the nest.

Input

The first line of the input contains two integers n and m — the numbers of rooms and nuts. They satisfy $1 \leq n \leq 2 \cdot 10^5$ and $0 \leq m < n$. Rooms are enumerated from 1 to n. The ant is initially in the room 1.

Each of the following $n-1$ lines has two integers x_i and y_i $(1 \leq i \leq n-1)$ indicating that a tunnel connects rooms numbered x_i and y_i. $1 \leq x_i \leq n$, $1 \leq y_i \leq n$, and $x_i \neq y_i$ hold. No two tunnels connect the same pair of rooms.

Each of the remaining m lines has an integer a_k $(1 \leq k \leq m, 2 \leq a_k \leq n)$ which indicates that the room numbered a_k has a nut in it. The numbers a_k's are all distinct.

Output

The output should be a single line containing a single integer, the number of rooms that the ant can reach and enter.

Examples

standard input	standard output
7 2	2
1 2	
2 3	
3 4	
4 5	
5 6	
5 7	
2	
4	
8 2	7
1 2	
2 3	
3 4	
2 5	
5 6	
6 7	
2 8	
2	
6	
7 3	2
1 2	
2 3	
3 4	
3 5	
4 6	
5 7	
2	
4	
5	
11 3	9
1 2	
2 3	
3 4	
2 5	
5 6	
6 7	
7 8	
6 9	
9 10	
6 11	
2	
3	
7	

Problem K. Kevin’s Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Kevin is playing a game on character strings. At the start of a game, a string of lowercase letters, called the target string, is given. Each of the players submits one string of lowercase letters, called a bullet string, of the specified length. The winner is the one whose bullet string marks the highest score.

The score of a bullet string is the sum of the points of all of its suffixes. When the bullet string is $b_1 b_2 \ldots b_n$, the score for its suffix s_k starting with the k-th character $(1 \le k \le n)$, $b_k b_{k+1} \ldots b_n$, is the length of its longest common prefix with the target string. That is, with the target string $t_1 t_2 \ldots t_m$, the score of s_k is p when $t_j = b_{k+j-1}$ for $1 \le j \le p$ and either $p = m, k + p - 1 = n$, or $t_{p+1} \ne b_{k+p}$ holds.

Write a program finds the highest achievable score for the given target string and the bullet length.

Input

The input consists of a single test case with two lines. The first line contains the non-empty target string of at most 2000 lowercase English letters. The second line contains the length of the bullet string, a positive integer not exceeding 2000.

Output

Output the highest achievable score for the given target string and the given bullet length.

Examples

standard input	standard output
ababc 6	10
aabaacaabaa 102	251