Bubble Cup 13 - Finals [Online Mirror, unrated, Div. 1]

A. Wakanda Forever

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

In the Kingdom of Wakanda, the 2020 economic crisis has made a great impact on each city and its surrounding area. Cities have made a plan to build a fast train rail between them to boost the economy, but because of the insufficient funds, each city can only build a rail with one other city, and they want to do it together.

Cities which are paired up in the plan will share the cost of building the rail between them, and one city might need to pay more than the other. Each city knows the estimated cost of building their part of the rail to every other city. One city can not have the same cost of building the rail with two different cities.

If in a plan, there are two cities that are not connected, but the cost to create a rail between them is lower for each of them than the cost to build the rail with their current pairs, then that plan is not acceptable and the collaboration won't go on. Your task is to create a suitable plan for the cities (pairing of the cities) or say that such plan doesn't exist.

Input

First line contains one integer N $(2 \leq N \leq 10^3)$ — the number of cities.

Each of the next N lines contains $N-1$ integers $A_{i,1}, A_{i,2}, \ldots, A_{i,i-1}, A_{i,i+1}, \ldots, A_{i,N-1}$ $(1 \leq A_{i,j} \leq 10^9)$ — where $A_{i,j}$ represents the cost for city i to build the rail to city j. Note that in each line $A_{i,i}$ is skipped.

Output

Output should contain N integers $O_1, O_2, \ldots, O_N$, where O_i represents the city with which city i should build the rail with, or -1 if it is not possible to find the stable pairing.

Examples

input
4 35 19 20 76 14 75 23 43 78 14 76 98
output
3 4 1 2

input
4 2 5 8 7 1 12 4 6 7 8 4 5
output
-1

B. Valuable Paper

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The pandemic is upon us, and the world is in shortage of the most important resource: toilet paper. As one of the best prepared nations for this crisis, BubbleLand promised to help all other world nations with this valuable resource. To do that, the country will send airplanes to other countries carrying toilet paper.

In BubbleLand, there are N toilet paper factories, and N airports. Because of how much it takes to build a road, and of course legal issues, every factory must send paper to only one airport, and every airport can only take toilet paper from one factory.

Also, a road can't be built between all airport-factory pairs, again because of legal issues. Every possible road has number d given, number of days it takes to build that road.

Your job is to choose N factory-airport pairs, such that if the country starts building all roads at the same time, it takes the least amount of days to complete them.

Input
The first line contains two integers N $(1 \leq N \leq 10^4)$ - number of airports/factories, and M $(1 \leq M \leq 10^5)$ - number of available pairs to build a road between.

On next M lines, there are three integers u, v $(1 \leq u, v \leq N)$, d $(1 \leq d \leq 10^9)$ - meaning that you can build a road between airport u and factory v for d days.

Output
If there are no solutions, output -1. If there exists a solution, output the minimal number of days to complete all roads, equal to maximal d among all chosen roads.

Example

input
3 5 1 2 1 2 3 2 3 3 3 2 1 4 2 2 5
output
4

C. Dušan's Railway

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

As you may already know, Dušan is keen on playing with railway models. He has a big map with cities that are connected with railways. His map can be seen as a graph where vertices are cities and the railways connecting them are the edges. So far, the graph corresponding to his map is a tree. As you already know, a tree is a connected acyclic undirected graph.

He is curious to find out whether his railway can be optimized somehow. He wants to add so-called *shortcuts*, which are also railways connecting pairs of cities. This shortcut will *represent* the railways in the unique path in the tree between the pair of cities it connects. Since Dušan doesn't like repeating the railways, he has also defined good paths in his newly obtained network (notice that after adding the shortcuts, his graph is no more a tree). He calls a path *good*, if no edge appears more than once, either as a regular railway edge or as an edge represented by some shortcut (Every shortcut in a good path has length 1, but uses up all the edges it represents - they can't appear again in that path). Having defined good paths, he defines *good distance* between two cities to be the length of the shortest good path between them. Finally, the *shortcutting diameter* of his network is the largest good distance between any two cities.

Now he is curious to find out whether it is possible to achieve shortcutting diameter less or equal than k, while adding as few shortcuts as possible.

Your solution should add no more than $10 \cdot \mathbf{n}$ shortcuts.

Input
The first line in the standard input contains an integer n $(1 \leq n \leq 10^4)$, representing the number of the cities in Dušan's railway map, and an integer k $(3 \leq k \leq n)$ representing the shortcutting diameter that he wants to achieve.

Each of the following $n - 1$ lines will contain two integers u_i and v_i $(1 \leq u_i, v_i \leq n, u_i \neq v_i)$, meaning that there is a railway between cities u_i and v_i.

Output
The first line of the output should contain a number t representing the number of the shortcuts that were added.

Each of the following t lines should contain two integers u_i and v_i, signifying that a shortcut is added between cities u_i and v_i.

Example

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

```
3 7
3 5
3 6
3 1
7 9
7 10
7 4
7 5
```

Note

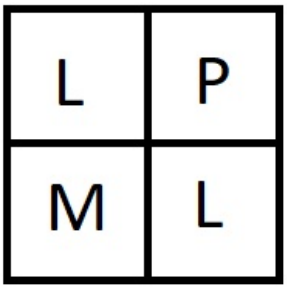
Notice that adding a shortcut between all cities and city 1 will make a graph theoretic diameter become 2. On the other hand, the paths obtained that way might not be good, since some of the edges might get duplicated. In the example, adding a shortcut between all cities and city 1 doesn't create a valid solution, because for cities 5 and 10 the path that uses shortcuts 5-1 and 1-10 is not valid because it uses edges 1-2, 2-3, 3-4, 4-5 twice.

D. Does anyone else hate the wind?

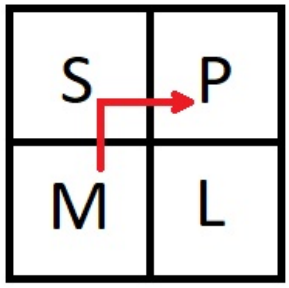
time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Mark and his crew are sailing across the sea of Aeolus (in Greek mythology Aeolus was the keeper of the winds). They have the map which represents the NxM matrix with land and sea fields and they want to get to the port (the port is considered as sea field). They are in a hurry because the wind there is very strong and changeable and they have the food for only K days on the sea (that is the maximum that they can carry on the ship). John, the guy from Mark's crew, knows how to predict the direction of the wind on daily basis for W days which is enough time for them to reach the port or to run out of the food. Mark can move the ship in four directions (north, east, south, west) by one field for one day, but he can also stay in the same place. Wind can blow in four directions (north, east, south, west) or just not blow that day. The wind is so strong at the sea of Aeolus that it moves the ship for one whole field in the direction which it blows at. The ship's resulting movement is the sum of the ship's action and wind from that day. Mark must be careful in order to keep the ship on the sea, the resulting movement must end on the sea field and there must be a 4-connected path through the sea from the starting field. A 4-connected path is a path where you can go from one cell to another only if they share a side.

For example in the following image, the ship can't move to the port as there is no 4-connected path through the sea.



In the next image, the ship can move to the port as there is a 4-connected path through the sea as shown with the red arrow. Furthermore, the ship can move to the port in one move if one of the following happens. Wind is blowing east and Mark moves the ship north, or wind is blowing north and Mark moves the ship east. In either of these scenarios the ship will end up in the port in one move.



Mark must also keep the ship on the map because he doesn't know what is outside. Lucky for Mark and his crew, there are T fish shops at the sea where they can replenish their food supplies to the maximum, but each shop is working on only one day. That means that Mark and his crew must be at the shop's position on the exact working day in order to replenish their food supplies. Help Mark to find the minimum of days that he and his crew need to reach the port or print -1 if that is impossible with the food supplies that they have.

Input

First line contains two integer numbers N and M ($1 \leq N, M \leq 200$) - representing the number of rows and number of columns of the map.

Second line contains three integers, K ($0 \leq K \leq 200$) which is the number of days with the available food supplies, T ($0 \leq T \leq 20$) which is the number of fields with additional food supplies and W ($0 \leq W \leq 10^6$) which is the number of days with wind information.

Next is the NxM char matrix filled with the values of 'L', 'S', 'P' or 'M'. 'L' is for the land and 'S' is for the sea parts. 'P' is for the port field and 'M' is the starting field for the ship.

Next line contains W chars with the wind direction information for every day. The possible inputs are 'N' - north, 'S' - south, 'E' - east, 'W' - west and 'C' - no wind. If Mark's crew can reach the port, it is guaranteed that they will not need more than W days to reach it.

In the end there are T lines with the food supplies positions. Each line contains three integers, Y_i and X_i ($0 \le Y_i < N, 0 \le X_i < M$) representing the coordinates (Y is row number and X is column number) of the food supply and F_i ($0 \le F_i \le 10^6$) representing the number of days from the starting day on which the food supply is available.

Output

One integer number representing the minimal days to reach the port or -1 if that is impossible.

Examples

input
3 3 5 2 15 M S S S S S S S P S W N N N N N N N N N N N 2 1 0 1 2 0
output
-1

input
3 3 5 2 15 M S S S S S S S P S E N N N N N N N N N N N 2 1 0 1 2 0
output
2

input
5 5 4 1 15 M S S S S S S S S L S S S L L S S S S S S S S S P C C C C S S E E C C C C C C C 0 1 4
output
8

E. 5G Antenna Towers

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

After making a strategic plan with carriers for expansion of mobile network throughout the whole country, the government decided to cover rural areas with the last generation of 5G network.

Since 5G antenna towers will be built in the area of mainly private properties, the government needs an easy way to find information about landowners for each **property** partially or fully contained in the planned building area.

The planned building area is represented as a rectangle with sides $width$ and $height$.

Every 5G antenna tower occupies a **circle** with a center in (x, y) and radius r.

There is a database of Geodetic Institute containing information about each property. Each property is defined with its **identification number** and **polygon** represented as an array of (x, y) points in the counter-clockwise direction.

Your task is to build an IT system which can handle queries of type (x, y, r) in which (x, y) represents a circle center, while r represents its radius. The IT system should return the **total area** of properties that need to be acquired for the building of a tower so that the government can estimate the price. Furthermore, the system should return a list of **identification numbers** of these properties (so that the owners can be contacted for land acquisition).

A property needs to be acquired if the circle of the antenna tower is intersecting or touching it.

Input
The first line contains the size of the building area as double values $width$, $height$, and an integer n — the number of properties in the database.

Each of the next n lines contains the description of a single property in the form of an integer number v ($3 \le v \le 40$) — the number of points that define a property, as well as $2 * v$ double numbers — the coordinates (x, y) of each property point. Line i ($0 \le i \le n - 1$) contains the information for property with id i.

The next line contains an integer q — the number of queries.

Each of the next q lines contains double values x, y, r — the coordinates of an antenna circle center (x, y) and its radius r.

$1 \le n * q \le 10^6$

Output
For each of the q queries, your program should output a line containing the total area of all the properties that need to be acquired, an integer representing the number of such properties, as well as the list of ids of these properties (separated by blank characters, arbitrary order).

Example

input
10 10 3 4 2 2 3 2 3 3 2 3 3 3.5 2 4.5 2 4.5 3 4 7 8 7.5 8.5 8 8 7.5 9 5 2 3.5 0.5 3.3 2 0.4 5 2.5 0.5 7.5 8.5 0.5 3 7 0.5
output
1.000000 1 0 1.500000 2 0 1 0.500000 1 1 0.250000 1 2 0.000000 0

Note
You can assume that the land not covered with properties (polygons) is under the government's ownership and therefore doesn't need to be acquired. Properties do not intersect with each other.

Precision being used for solution checking is 10^{-4}.

F. Coins

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

A famous gang of pirates, Sea Dogs, has come back to their hideout from one of their extravagant plunders. They want to split their treasure fairly amongst themselves, that is why You, their trusted financial advisor, devised a game to help them:

All of them take a sit at their round table, some of them with the golden coins they have just stolen. At each iteration of the game if one of them has equal or more than 2 coins, he is eligible to the splitting and he gives one coin to each pirate sitting next to him. If there are more candidates (pirates with equal or more than 2 coins) then **You** are the one that chooses which **one** of them will do the splitting in that iteration. The game ends when there are no more candidates eligible to do the splitting.

Pirates can call it a day, only when the game ends. Since they are beings with a finite amount of time at their disposal, they would prefer if the game that they are playing can end after finite iterations, and if so, they call it a **good game**. On the other hand, if no matter how **You** do the splitting, the game cannot end in finite iterations, they call it a **bad game**. Can You help them figure out before they start playing if the game will be good or bad?

Input
The first line of input contains two integer numbers n and k ($1 \le n \le 10^9$, $0 \le k \le 2 \cdot 10^5$), where n denotes total number of pirates and k is the number of pirates that have any coins.

The next k lines of input contain integers a_i and b_i ($1 \le a_i \le n$, $1 \le b_i \le 10^9$), where a_i denotes the index of the pirate sitting at the round table (n and 1 are neighbours) and b_i the total number of coins that pirate a_i has at the start of the game.

Output
Print 1 if the game is a **good game**: There is a way to do the splitting so the game ends after finite number of iterations.

Print -1 if the game is a **bad game**: No matter how You do the splitting the game does not end in finite number of iterations.

Examples

input
4 2 1 2 2 2
output
1

input
6 2 2 3 4 1
output
1

input
3 2 1 1 2 2
output
-1

Note
In the third example the game has no end, because You always only have only one candidate, after whose splitting you end up in the same position as the starting one.

G. Growing flowers

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Sarah has always been a lover of nature, and a couple of years ago she saved up enough money to travel the world and explore all the things built by nature over its lifetime on earth. During this time she visited some truly special places which were left untouched for centuries, from watching icebergs in freezing weather to scuba-diving in oceans and admiring the sea life, residing unseen. These experiences were enhanced with breathtaking views built by mountains over time and left there for visitors to see for years on end. Over time, all these expeditions took a toll on Sarah and culminated in her decision to settle down in the suburbs and live a quiet life.

However, as Sarah's love for nature never faded, she started growing flowers in her garden in an attempt to stay connected with nature. At the beginning she planted only blue orchids, but over time she started using different flower types to add variety to her collection of flowers. This collection of flowers can be represented as an array of N flowers and the i-th of them has a type associated with it, denoted as A_i. Each resident, passing by her collection and limited by the width of his view, can only see K contiguous flowers at each moment in time. To see the whole collection, the resident will look at the first K contiguous flowers $A_1, A_2, \ldots, A_K$, then shift his view by one flower and look at the next section of K contiguous flowers $A_2, A_3, \ldots, A_{K+1}$ and so on until they scan the whole collection, ending with section $A_{N-K+1}, \ldots, A_{N-1}, A_N$.

Each resident determines the beautiness of a section of K flowers as the number of distinct flower types in that section. Furthermore, the **beautiness** of the whole collection is calculated by summing the beautiness values of each contiguous section. Formally, beautiness B_i of a section starting at the i-th position is calculated as $B_i = distinct(A_i, A_{i+1}, \ldots, A_{i+K-1})$, and **beautiness** of the collection B is calculated as $B = B_1 + B_2 + \ldots + B_{N-K+1}$.

In addition, as Sarah wants to keep her collection of flowers have a fresh feel, she can also pick two points L and R, dispose flowers between those two points and plant new flowers, all of them being the same type.

You will be given Q queries and each of those queries will be of the following two types:

1. You will be given three integers L, R, X describing that Sarah has planted flowers of type X between positions L and R inclusive. Formally collection is changed such that $A[i] = X$ for all i in range $[L..R]$.
2. You will be given integer K, width of the resident's view and you have to determine the beautiness value B resident has associated with the collection

For each query of second type print the result – beautiness B of the collection.

Input
First line contains two integers N and Q $(1 \leq N, Q \leq 10^5)$ — number of flowers and the number of queries, respectively.

The second line contains N integers $A_1, A_2, \ldots, A_N$ $(1 \leq A_i \leq 10^9)$ — where A_i represents type of the i-th flower.

Each of the next Q lines describe queries and start with integer $T \in \{1, 2\}$.

- If $T = 1$, there will be three more integers in the line L, R, X $(1 \leq L, R \leq N;\ 1 \leq X \leq 10^9)$ — L and R describing boundaries and X describing the flower type

- If $T = 2$, there will be one more integer in the line K $(1 \le K \le N)$ — resident's width of view

Output
For each query of the second type print the beautiness B of the collection.

Example

input
5 5 1 2 3 4 5 2 3 1 1 2 5 2 4 1 2 4 5 2 2
output
9 6 4

Note
Let's look at the example.

Initially the collection is $[1, 2, 3, 4, 5]$. In the first query $K = 3$, we consider sections of three flowers with the first being $[1, 2, 3]$. Since beautiness of the section is the number of distinct flower types in that section, $B_1 = 3$. Second section is $[2, 3, 4]$ and $B_2 = 3$. Third section is $[3, 4, 5]$ and $B_3 = 3$, since the flower types are all distinct. The beautiness value resident has associated with the collection is $B = B_1 + B_2 + B_3 = 3 + 3 + 3 = 9$.

After the second query, the collection becomes $[5, 5, 3, 4, 5]$.

For the third query $K = 4$, so we consider sections of four flowers with the first being $[5, 5, 3, 4]$. There are three distinct flower types $[5, 3, 4]$ in this section, so $B_1 = 3$. Second section $[5, 3, 4, 5]$ also has 3 distinct flower types, so $B_2 = 3$. The beautiness value resident has associated with the collection is $B = B_1 + B_2 = 3 + 3 = 6$

After the fourth query, the collection becomes $[5, 5, 5, 5, 5]$.

For the fifth query $K = 2$ and in this case all the four sections are same with each of them being $[5, 5]$. Beautiness of $[5, 5]$ is 1 since there is only one distinct element in this section $[5]$. Beautiness of the whole collection is $B = B_1 + B_2 + B_3 + B_4 = 1 + 1 + 1 + 1 = 4$

H. Virus

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

In Bubbleland a group of special programming forces gets a top secret job to calculate the number of potentially infected people by a new unknown virus. The state has a population of n people and every day there is new information about new contacts between people. The job of special programming forces is to calculate how many contacts in the last k days a given person had.

The new virus has an incubation period of k days, and after that time people consider as non-infectious. Because the new virus is an extremely dangerous, government mark as suspicious everybody who had direct or indirect contact in the last k days, independently of the order of contacts.

This virus is very strange, and people can't get durable immunity.

You need to help special programming forces to calculate the number of suspicious people for a given person (number of people who had contact with a given person).

There are 3 given inputs on beginning n where n is population, q number of queries, k virus incubation time in days. Each query is one of three types:

1. (x, y) person x and person y met that day ($x \neq y$).
2. (z) return the number of people in contact with z, counting himself.
3. The end of the current day moves on to the next day.

Input
The first line of input contains three integers n ($1 \le n \le 10^5$) the number of people in the state, q ($1 \le q \le 5 \times 10^5$) number of queries and k ($1 \le k \le 10^5$) virus incubation time in days.

Each of the next q lines starts with an integer t ($1 \le t \le 3$) the type of the query.

A pair of integers x and y ($1 \le x, y \le n$) follows in the query of the first type ($x \neq y$).

An integer i ($1 \le i \le n$) follows in the query of the second type.

Query of third type does not have the following number.

Output

For the queries of the second type print on a separate line the current number of people in contact with a given person.

Examples

input
5 12 1 1 1 2 1 1 3 1 3 4 2 4 2 5 3 2 1 1 1 2 1 3 2 2 1 3 2 1
output
4 1 1 3 1

input
5 12 2 1 1 2 1 1 3 1 3 4 2 4 2 5 3 2 1 1 1 2 1 3 2 2 1 3 2 1
output
4 1 4 4 3

input
10 25 2 1 9 3 2 5 1 1 3 1 3 1 2 2 1 8 3 1 5 6 3 1 9 2 1 8 3 2 9 1 3 1 2 5 1 6 4 3 3 2 4 3 1 10 9 1 1 7 3 2 2 3 1 5 6 1 1 4
output
1 1 5 2 1 1

Note

Pay attention if persons 1 and 2 had contact first day and next day persons 1 and 3 had contact, for k>1 number of contacts of person 3 is 3(persons:1,2,3).

I. Lookup Tables

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

John has Q closed intervals of consecutive $2K$-bit numbers $[l_i, r_i]$ and one 16-bit value v_i for each interval. ($0 \leq i < Q$)

John wants to implement a function F that maps $2K$-bit numbers to 16-bit numbers in such a way that inputs from each interval are mapped to that interval's value. In other words:

$$F(x) = v_i, \text{ for every } 0 \leq i < Q \text{ , and every } x \in [l_i, r_i]$$

The output of F for other inputs is unimportant.

John wants to make his implementation of F fast so he has decided to use lookup tables. A single $2K$-bit lookup table would be too large to fit in memory, so instead John plans to use two K-bit lookup tables, LSBTable and MSBTable. His implementation will look like this:

$$F(x) = \text{LSBTable}[\text{lowKBits}(x)] \ \& \ \text{MSBTable}[\text{highKBits}(x)]$$

In other words it returns the "bitwise and" of results of looking up the K least significant bits in LSBTable and the K most significant bits in MSBTable.

John needs your help. Given K, Q and Q intervals $[l_i, r_i]$ and values v_i, find any two lookup tables which can implement F or report that such tables don't exist.

Input

The first line contains two integers K and Q ($1 <= K <= 16$, $1 <= Q <= 2 \cdot 10^5$).

Each of the next Q lines contains three integers l_i, r_i and v_i. ($0 \leq l_i \leq r_i < 2^{2K}$, $0 \leq v_i < 2^{16}$).

Output

On the first line output "possible" (without quotes) if two tables satisfying the conditions exist, or "impossible" (without quotes) if they don't exist.

If a solution exists, in the next $2 \cdot 2^K$ lines your program should output all values of the two lookup tables (LSBTable and MSBTable) it found. When there are multiple pairs of tables satisfying the conditions, your program may output any such pair.

On lines $1 + i$ output $\text{LSBTable}[i]$. ($0 \leq i < 2^K$, $0 \leq \text{LSBTable}[i] < 2^{16}$).

On lines $1 + 2^K + i$ output $\text{MSBTable}[i]$. ($0 \leq i < 2^K$, $0 \leq \text{MSBTable}[i] < 2^{16}$).

Examples

input
1 2 0 2 1 3 3 3
output
possible 1 3 1 3

input
2 4 4 5 3 6 7 2 0 3 0 12 13 1
output
possible 3 3 2 2 0 3 0 1

input
2 3 4 4 3 5 6 2 12 14 1

output
impossible

Note
A closed interval $[a, b]$ includes both a and b.

In the first sample, tables $\mathrm{LSBTable} = [1, 3]$ and $\mathrm{MSBTable} = [1, 3]$ satisfy the conditions:
$F[0] = \mathrm{LSBTable}[0]\&\mathrm{MSBTable}[0] = 1\&1 = 1$, $F[1] = \mathrm{LSBTable}[1]\&\mathrm{MSBTable}[0] = 3\&1 = 1$,
$F[2] = \mathrm{LSBTable}[0]\&\mathrm{MSBTable}[1] = 1\&3 = 1$, $F[3] = \mathrm{LSBTable}[1]\&\mathrm{MSBTable}[1] = 3\&3 = 3$.

In the second sample, tables $\mathrm{LSBTable} = [3, 3, 2, 2]$ and $\mathrm{MSBTable} = [0, 3, 0, 1]$ satisfy all the conditions.

In the third sample there are no two lookup tables which can satisfy the conditions.

J. Bubble Cup hypothesis

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

The Bubble Cup hypothesis stood unsolved for 130 years. Who ever proves the hypothesis will be regarded as one of the greatest mathematicians of our time! A famous mathematician Jerry Mao managed to reduce the hypothesis to this problem:

Given a number m, how many polynomials P with coefficients in set $\{0, 1, 2, 3, 4, 5, 6, 7\}$ have: $P(2) = m$?

Help Jerry Mao solve the long standing problem!

Input
The first line contains a single integer t $(1 \le t \le 5 \cdot 10^5)$ - number of test cases.

On next line there are t numbers, m_i $(1 \le m_i \le 10^{18})$ - meaning that in case i you should solve for number m_i.

Output
For each test case i, print the answer on separate lines: number of polynomials P as described in statement such that $P(2) = m_i$, modulo $10^9 + 7$.

Example

input
2 2 4
output
2 4

Note
In first case, for $m = 2$, polynomials that satisfy the constraint are x and 2.

In second case, for $m = 4$, polynomials that satisfy the constraint are x^2, $x + 2$, $2x$ and 4.

K. Lonely Numbers

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

In number world, two different numbers are friends if they have a lot in common, but also each one has unique perks.

More precisely, two different numbers a and b are friends if $gcd(a, b)$, $\frac{a}{gcd(a,b)}$, $\frac{b}{gcd(a,b)}$ can form sides of a triangle.

Three numbers a, b and c can form sides of a triangle if $a + b > c$, $b + c > a$ and $c + a > b$.

In a group of numbers, a number is lonely if it doesn't have any friends in that group.

Given a group of numbers containing all numbers from $1, 2, 3, \ldots, n$, how many numbers in that group are lonely?

Input
The first line contains a single integer t $(1 \le t \le 10^6)$ - number of test cases.

On next line there are t numbers, n_i $(1 \le n_i \le 10^6)$ - meaning that in case i you should solve for numbers $1, 2, 3, \ldots, n_i$.

Output
For each test case, print the answer on separate lines: number of lonely numbers in group $1, 2, 3, \ldots, n_i$.

Example

input
3 1 5 10
output
1 3 3

Note

For first test case, 1 is the only number and therefore lonely.

For second test case where $n = 5$, numbers 1, 3 and 5 are lonely.

For third test case where $n = 10$, numbers 1, 5 and 7 are lonely.

L. Light switches

time limit per test: 3 seconds
memory limit per test: 1024 megabytes
input: standard input
output: standard output

Nikola owns a large warehouse which is illuminated by N light bulbs, numbered 1 to N. At the exit of the warehouse, there are S light switches, numbered 1 to S. Each switch swaps the on/off state for some light bulbs, so if a light bulb is off, flipping the switch turns it on, and if the light bulb is on, flipping the switch turns it off.

At the end of the day, Nikola wants to turn all the lights off. To achieve this, he will flip some of the light switches at the exit of the warehouse, but since Nikola is lazy, he wants to flip the _minimum_ number of switches required to turn all the lights off. Since Nikola was not able to calculate the minimum number of switches, he asked you to help him. During a period of D days, Nikola noted which light bulbs were off and which were on at the end of each day. He wants you to tell him the minimum number of switches he needed to flip to turn all the lights off for each of the D days or tell him that it's impossible.

Input

First line contains three integers, N, S and D ($1 \leq N \leq 10^3$, $1 \leq S \leq 30$, $1 \leq D \leq 10^3$) – representing number of light bulbs, the number of light switches, and the number of days respectively.

The next S lines contain the description of each light switch as follows: The first number in the line, C_i ($1 \leq C_i \leq N$), represents the number of light bulbs for which the on/off state is swapped by light switch i, the next C_i numbers (sorted in increasing order) represent the indices of those light bulbs.

The next D lines contain the description of light bulbs for each day as follows: The first number in the line, T_i ($1 \leq T_i \leq N$), represents the number of light bulbs which are on at the end of day i, the next T_i numbers (sorted in increasing order) represent the indices of those light bulbs.

Output

Print D lines, one for each day. In the i^{th} line, print the minimum number of switches that need to be flipped on day i, or -1 if it's impossible to turn all the lights off.

Example

input
4 3 4 2 1 2 2 2 3 1 2 1 1 2 1 3 3 1 2 3 3 1 2 4
output
2 2 3 -1

M. Milutin's Plums

time limit per test: 1 second
memory limit per test: 1024 megabytes
input: standard input
output: standard output

As you all know, the plum harvesting season is on! Little Milutin had his plums planted in an orchard that can be represented as an n by m matrix. While he was harvesting, he wrote the heights of all trees in a matrix of dimensions n by m.

At night, when he has spare time, he likes to perform various statistics on his trees. This time, he is curious to find out the height of his lowest tree. So far, he has discovered some interesting properties of his orchard. There is one particular property that he thinks is useful for finding the tree with the smallest heigh.

Formally, let $L(i)$ be the leftmost tree with the smallest height in the i-th row of his orchard. He knows that $L(i) \leq L(i+1)$ for all $1 \leq i \leq n-1$. Moreover, if he takes a submatrix induced by any subset of rows and any subset of columns, $L(i) \leq L(i+1)$ will hold for all $1 \leq i \leq n' - 1$, where n' is the number of rows in that submatrix.

Since the season is at its peak and he is short on time, he asks you to help him find the plum tree with minimal height.

Input
This problem is interactive.

The first line of input will contain two integers n and m, representing the number of rows and the number of columns in Milutin's orchard. It is guaranteed that $1 \leq n, m \leq 10^6$.

The following lines will contain the answers to your queries.

Output
Once you know have found the minimum value r, you should print ! r to the standard output.

Interaction
Your code is allowed to query for an entry (i, j) of a matrix (i.e. get the height of the tree which is in the i-th row and j-th column). The query should be formatted as ? i j, so that $1 \leq i \leq n$ and $1 \leq j \leq m$.

You may assume that the entries of the matrix will be integers between 1 and 10^9.

Your solution should use not more than $4 \cdot (n + m)$ queries.

This is an interactive problem. You have to use a `flush` operation right after printing each line. For example, in C++ you should use the function `fflush(stdout)`, in Java — `System.out.flush()`, in Pascal — `flush(output)` and in Python — `sys.stdout.flush()`.

Example

input
5 5 13 15 10 9 15 15 17 12 11 17 10 12 7 6 12 17 19 14 13 19 16 18 13 12 18
output

N. BubbleSquare Tokens

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

BubbleSquare social network is celebrating 13^{th} anniversary and it is rewarding its members with special edition BubbleSquare tokens. Every member receives one personal token. Also, two additional tokens are awarded to each member for every friend they have on the network. Yet, there is a twist – everyone should end up with different number of tokens from all their friends. Each member may return one received token. Also, each two friends may agree to each return one or two tokens they have obtained on behalf of their friendship.

Input
First line of input contains two integer numbers n and k ($2 \leq n \leq 12500$, $1 \leq k \leq 1000000$) - number of members in network and number of friendships.

Next k lines contain two integer numbers a_i and b_i ($1 \leq a_i, b_i \leq n$, $a_i \neq b_i$) - meaning members a_i and b_i are friends.

Output
First line of output should specify the number of members who are keeping their personal token.

The second line should contain space separated list of members who are keeping their personal token.

Each of the following k lines should contain three space separated numbers, representing friend pairs and number of tokens each of them gets on behalf of their friendship.

Examples

input
2 1 1 2
output

```
1
1
1 2 0
```

input
3 3 1 2 1 3 2 3
output
0 1 2 0 2 3 1 1 3 2

Note

In the first test case, only the first member will keep its personal token and no tokens will be awarded for friendship between the first and the second member.

In the second test case, none of the members will keep their personal token. The first member will receive two tokens (for friendship with the third member), the second member will receive one token (for friendship with the third member) and the third member will receive three tokens (for friendships with the first and the second member).