Problem A. Adjacent Rooks

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Professor Oak is preparing a problem for his students. The problem consists in counting the number of ways of placing n rooks on an $n \times n$ chessboard such that no rook is threatening another rook (that is, no two rooks are in the same column or in the same row).

However, this problem is too easy, so he has decided to add a twist. He only wants you to count the number of solutions in which there are exactly k pairs of rooks that are diagonally adjacent (that is, they are in neighboring columns and in neighboring rows). Can you solve it?

Output your answer modulo $10^9 + 7$.

Input

The first line contains a single integer t, the number of test cases ($1 \le t \le 5000$).

Each test case is given on a single line containing two integers n and k: ($1 \le n \le 1000$; $0 \le k \le n - 1$): the number of rooks and the number of pairs of rooks that must be diagonally adjacent.

Output

Print one integer: the number of ways to place n rooks such that no two are in the same row or column and such that there are exactly k pairs of diagonally adjacent rooks. The answer must be expressed modulo $10^9 + 7$.

Example

standard input	standard output
5 1 0 2 0 3 1 3 2 4 2	1 0 4 2 10

Problem B. Beautiful Permutation

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

A permutation $a_0, a_1, \ldots, a_{n-1}$ of $0, 1, \ldots, n-1$ is said to be *beautiful* if the sequence $b_0, \ldots, b_{n-1}$ defined as $b_i = |a_i - i|$ is also a permutation of $0, \ldots, n-1$.

Given n, construct a beautiful permutation of n elements or determine that it does not exist.

Input

The first line contains a single integer n ($1 \le n \le 10^6$): the size of the permutation.

Output

If there is no beautiful permutation of n elements, output a single line with the word "NO".

Otherwise, on the first line, print "YES", and on the second line, print n space-separated integers $a_0, \ldots, a_{n-1}$: the beautiful permutation. If there are multiple beautiful permutations, print any one of them.

Examples

standard input	standard output
4	YES 3 0 2 1
3	NO
1	YES 0

Problem C. Cartesian MST

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Let G and H be two weighted undirected simple graphs. We define the *cartesian product* of the two graphs, $G\square H$, as the graph whose vertex set is the cartesian set product of the vertex sets of the two graphs $V(G) \times V(H)$ and in which there is an edge between vertices (u_1, v_1) and (u_2, v_2) if and only if:

- $v_1 = v_2$ and there is an edge (u_1, u_2) in G. In this case, the edge$((u_1, v_1), (u_2, v_2))$ in $G\square H$ has the same weight as the edge (u_1, u_2) in G.
- or $u_1 = u_2$ and there is an edge (v_1, v_2) in H. In this case, the edge$((u_1, v_1), (u_2, v_2))$ in $G\square H$ has the same weight as the edge (v_1, v_2) in H.

You are given two connected graphs G and H. Compute the total weight of the minimum spanning tree of $G\square H$.

Input

The first line contains four integers n_1, m_1, n_2, m_2 ($2 \le n_1, n_2 \le 10^5$; $1 \le m_1, m_2 \le 10^5$): the number of vertices of G, the number of edges of G, the number of vertices of H, and the number of edges of H, respectively.

Each of the next m_1 lines contains three integers u_i, v_i, w_i ($0 \le u_i, v_i \le n_1 - 1$; $1 \le w_i \le 10^8$), describing an edge of G between vertices u_i and v_i with weight w_i.

Each of the next m_2 lines contains three integers u_i, v_i, w_i ($0 \le u_i, v_i \le n_2 - 1$; $1 \le w_i \le 10^8$), describing an edge of H between vertices u_i and v_i with weight w_i.

It is guaranteed that graphs G and H are simple and connected. Recall that a graph is *simple* if there are no edges between a vertex and itself, and there is at most one edge between any two vertices.

Output

Output one integer: the weight of the minimum spanning tree of $G\square H$.

Example

standard input	standard output
4 4 3 2 0 1 3 1 2 2 2 3 2 3 0 5 0 1 1 1 2 1	15

Problem D. Display of Springs

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

Elater is a great magician. His most famous show is "Elater's Super Spectacular Display of Springs". The show consists in the following:

There are n elastic springs in a line attached to the ceiling. Spring i is attached at height h_i and has a *stiffness constant* k_i. If we attach a weight of mass w to the lower end of the i-th spring, the weight will descend to a height H given by the formula:

$$H = h_i - \frac{w}{k_i}.$$

Elater will take questions from people in the audience. When asked about a positive integer w, Elater will be able to pick the spring which, if a weight of mass w is attached to its lower end, will descend to a height lower (closer to the floor) than all the other springs (in case of a tie, he will be able to pick one of the springs with the lowest descend height). To accomplish such an amazing feat, Elater has the help of his dear assistant, Hooke.

Before the show, Hooke has some time to do measurements on the springs. He can not directly measure the values of h_i and k_i, but he can choose two springs a and b and a mass w, try attaching a weight of mass w to the two springs, and see which of them goes closer to the floor with that weight. Before the show, he has time to do up to 20 000 such measurements. Also, after each question, Elater can distract the audience for a while, so that Hooke can do up to 20 more measurements before inconspicuously whispering the answer to Elater.

Can you help Hooke make the show successful? Assume that the springs are attached high enough, and the masses are small enough, so that the weights never reach the floor during any possible measurement. Also note that the weights are removed after each measurement.

Interaction Protocol

First, you must read a single line with the integer n ($2 \le n \le 500$) from the input.

To make a measurement, you must print a single line formatted as "`?` a b w", where a and b ($0 \le a, b \le n - 1$) are the indices of the two springs you want to use, and w ($1 \le w \le 10^5$) is the integer weight you want to put in both springs. After that, you must read from the input a single line with the answer, which will be the word "`FIRST`" if spring a reaches lower height, "`SECOND`" if spring b reaches lower height, or "`EQUAL`" if both reach the same height.

During the first stage, you can make at most 20 000 measurements. After doing all the measurements of the first stage, print a single line with a single character "`!`". After that, you will be given one or more questions.

Each question is given on a single line formatted as "`QUESTION` w" where w is the integer weight Elater is asked about ($1 \le w \le 10^5$). After reading each question, you can do at most 20 measurements, in the same format as in the first stage. Once you have done the measurements, output a single line formatted as "`!` i", where i ($0 \le i \le n - 1$) is the index of the spring that is the answer to the question. If there are multiple springs that reach the same minimum height, you may print any of them.

After all questions, instead of the next one, you will get a single line consisting of the word "`FINISH`". It means there are no more questions and your program must finish. The total number of questions will be at least 1 and at most 1000.

Don't forget to flush the output after printing each line!

In each test, the values of h_i, k_i, and the questions are all fixed in advance.

Example

standard input	standard output
3 SECOND QUESTION 2 SECOND FIRST QUESTION 6 SECOND SECOND FINISH	 ? 0 1 1 ! ? 0 1 2 ? 1 2 2 ! 1 ? 0 1 6 ? 1 2 6 ! 2

Problem E. Even Intervals

Input file:	*standard input*
Output file:	*standard output*
Time limit:	20 seconds
Memory limit:	1024 mebibytes

You are given an array with n pairwise different values: $A = [a_0, a_1, \ldots, a_{n-1}]$. We define the sorted subarray of A starting at l and ending at r as the array that we obtain after sorting $[a_l, a_{l+1}, \ldots, a_r]$. For example, if we are given the array $[0, 2, 14, 6, 8, 10]$, the sorted subarray starting at 1 and ending at 4 would be the array that we would get after sorting $[2, 14, 6, 8]$, that is, the array $[2, 6, 8, 14]$.

You are given q queries, each one consists of two integers, l and r. For each query, print the sum of the values in the even positions of the sorted subarray of A starting at l and ending at r. Here, we assume that all arrays are indexed starting from 0.

For example, consider the array $[0, 2, 14, 6, 8, 10]$ and the query $(1, 4)$. The subarray starting at 1 and ending at 4 is just the array $[2, 14, 6, 8]$. Thus, the sorted subarray starting at 1 and ending at 4 is the array $[2, 6, 8, 14]$. Now we have to sum the values in even positions, that is, $2 + 8 = 10$.

Print the answers modulo $10^9 + 7$.

Input

The first line contains two integers n and q ($1 \le n \le 5 \cdot 10^4$; $1 \le q \le 2 \cdot 10^5$): the number of elements in the array and the number of queries.

The second line contains n integers $a_0, a_1, \ldots, a_{n-1}$ ($0 \le a_i \le 10^9$; a_i are pairwise different), the elements of the array.

Finally, each of the next q lines contains two integers l and r ($0 \le l \le r < n$): the starting and ending points of the sorted subarray we are considering.

Output

For each query, print a line with the sum of the elements in even positions of the sorted subarray starting at l and ending at r modulo $10^9 + 7$.

Examples

standard input	standard output
5 5 2 4 10 16 6 0 2 1 3 0 3 2 3 0 4	12 20 12 10 24
8 8 38 20 76 96 74 18 66 92 0 5 3 6 1 2 2 7 0 6 2 2 1 6 5 5	132 92 20 184 226 76 160 18

Problem F. Friendship Circles

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

Let $p_0, p_1, \ldots, p_{n-1}$ be n points in the plane. We say that two points are *friends* if one can draw a circle that contains both points in its interior and all the other $n-2$ points in its exterior. Print the indices of the points that are friends with p_0.

It is guaranteed that there is no circumference containing p_0 and three or more other points. It is also guaranteed that there is no line containing p_0 and two or more other points.

Input

The first line contains an integer t, the number of test cases ($1 \le t \le 10^4$).

Each test case starts with a line containing an integer n ($2 \le n \le 10^5$), the number of points. It is followed by n lines, each one containing two integers x_i and y_i ($-10^9 \le x_i, y_i \le 10^9$): the coordinates of the i-th point.

The tests are not explicitly targeting precision issues. In particular, it is guaranteed that, if we moved p_0 by a distance of at most 10^{-6} units in any direction, the answer would remain the same.

The total number of points in all test cases does not exceed 10^5.

Output

For each test case, print a line containing one integer m, the number of friends of p_0, followed by m integers: the indices of the friends of p_0 in lexicographical order.

Example

standard input	standard output
2 4 1 0 3 1 3 -2 7 0 5 0 0 -2 -1 2 2 -2 10 -1 11	2 1 2 3 1 2 3

Problem G. Game on a Tree

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

Some friends, numbered from 0 to $m-1$, are playing a game. There are cards of c different colors, where colors are numbered from 0 to $c-1$. Each card of color i has a fixed value p_i. At the beginning of the game, each player chooses a certain amount of cards, without taking more than one card of the same color (though different friends may choose cards of the same color). We define the *score* of a deck as the GCD (greatest common divisor) of the values of its cards. Notice that an empty deck has a score of 0 points. The game will be played on an undirected graph which is a tree consisting of n nodes, numbered from 0 to $n-1$. Each node v contains a card with color c_v, and different nodes may have cards of the same color.

The game has m rounds. In round i, the i-th friend gets to play. First, he chooses two different vertices v and u. Then, for each node v_i on the path from v to u (including v and u), each player (including himself) draws a card of color c_{v_i}. If they already had a card of color c_{v_i}, they will discard both cards. After doing this, player i gets as many points as the sum of the points of each player's deck (including himself). Finally, he chooses a node of the graph and draws a new card from the pile. Then he changes the node's card for the card he has just drawn (and discards the old card), ending his round.

Given the information of each round, print how many points each friend will have at the end of the game modulo 10^9+7.

Input

The first line contains three integers m, c, n ($2 \le n, m \le 10^5$; $1 \le c \le 20$): the number of friends (and hence, rounds), the number of different colors of the cards, and the number of nodes the tree will have.

The second line contains c integers $p_0, p_1, \ldots, p_{c-1}$ ($0 \le p_i \le 10^9$), where p_i is the value of a card of color i.

Each of the next m lines starts with an integer x ($0 \le x \le c$), the number of cards the i-th player starts with. It is followed by x integers $y_0, y_1, \ldots, y_{x-1}$ ($0 \le y_i < c$): the colors of all the player's cards.

The next line contains n integers $c_0, c_1, \ldots, c_{n-1}$ ($0 \le c_i < c$), the color of the card in each node of the graph.

Then $n-1$ lines follow, each one containing two integers u and v ($0 \le u, v < n$), meaning that there is an edge between nodes u and v. It is guaranteed that these edges form a tree.

Finally, each of the last m lines has four integers u, v, w, y ($0 \le u, v, w < n$; $u \ne v$; $0 \le y < c$) that encode the information about each of the m rounds: the two chosen nodes (u and v) and the node w that will have its card changed for one with color y.

Output

Print one line with m integers: $points_0, points_1, \ldots, points_{m-1}$, where $points_i$ is the number of points the i-th friend ends up with, modulo 10^9+7.

Examples

standard input	standard output
3 4 7 0 2 3 6 2 1 2 3 0 2 3 2 0 1 2 1 2 3 0 1 1 0 1 2 0 3 0 1 4 6 4 4 5 5 2 4 1 6 1 1 2 4 2 0 3	6 4 9
7 4 8 28 49 88 49 1 1 3 2 0 3 0 1 3 0 0 1 0 2 2 2 2 3 0 1 2 0 2 0 3 2 6 0 4 3 7 7 1 2 5 1 4 4 1 3 6 4 2 7 6 4 2 6 2 2 2 4 2 7 0 2 3 5 3 5 1 4 1	207 13 121 253 13 253 106

Problem H. Hackerman

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

You, Mr. Hackerman, have gained access to the servers of a secure messaging app. They use a new ciphering method based on the difficulty of factoring the numbers that are products of three big primes.

You are reading the source code and found out how they actually generate these numbers. They define three recurrences:

- $x_n = (11 \cdot x_{n-1} + 7) \bmod 611\,953 \quad \forall n > 0$
- $y_n = (13 \cdot y_{n-1} + 5) \bmod 746\,773 \quad \forall n > 0$
- $z_n = (53 \cdot z_{n-1} + 3) \bmod 882\,389 \quad \forall n > 0$

The seeds x_0, y_0, z_0 seem to be stored in a secure file.

Now they get the three primes p_k, q_k, r_k for the k-th user:

- p_k is x_k-th number in a secure file `X.axx` (all its numbers are different primes with exactly 31 decimal digits).
- q_k is y_k-th number in a secure file `Y.axx` (all its numbers are different primes with exactly 32 decimal digits).
- r_k is z_k-th number in a secure file `Z.axx` (all its numbers are different primes with exactly 33 decimal digits).

Then, they compute the public key $n_k = p_k \cdot q_k \cdot r_k$.

You have temporary access to the public key database, and you can query up to 5 public keys for any user.

You are given two integers u and v. You have to intercept communications between the u-th user and the v-th user. For this task, you will need to compute $p_u, q_u, r_u, p_v, q_v, r_v$; having these values will allow you to completely manipulate communication.

For the sake of input/output simplicity, we ask you to output $p_u + q_u + r_u + p_v + q_v + r_v$ as your answer.

Interaction Protocol

First, you must read a line containing two integers u and v ($0 \le u, v < 7 \cdot 10^{12}$).

To ask for the public key (n_k) of the k-th user, print "`?` k", where k is the user index. Remember that you can do at most 5 queries of this type.

The first user has index 0, and there are exactly $7 \cdot 10^{12}$ users, so make sure $0 \le k < 7 \cdot 10^{12}$ in all your questions.

When you have computed the solution, print "`!` a", where $a = p_u + q_u + r_u + p_v + q_v + r_v$. Remember it's safer to stop your code after this query.

Don't forget to flush the output after printing each line!

The files `X.axx`, `Y.axx`, and `Z.axx` are the same for the whole problem.

Example

standard input	standard output
10 20 192279309409462992645482090330404758368400469722499925076043266903464961794187094077107243967491 27484854406533716638162995259016486377602039494141055337350245326304213427822762176892360055761 7 619917161121620915718543801031971411330712020624376365924343965254284332847752540506954141582 03	? 10 ? 20 ? 3 ! 1188670725123074098790368447122696

Note

Solution to the example ($u = 10$ and $v = 20$):

$p_{10} = 67457197281134847949206967678 81$

$q_{10} = 54398126832702965410665141463513$

$r_{10} = 52398676217202370046677422543 0947$

$p_{20} = 68990370853239001493839571795 69$

$q_{20} = 76607972465670150189802211467309$

$r_{20} = 52003310683923989777882221481 3477$

$p_{10} + q_{10} + r_{10} + p_{20} + q_{20} + r_{20} = 1188670725123074098790368447122696$

Problem I. Interesting Scoring Systems

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Score in chess tournaments is a controversial topic. Abel likes the classical system: 2 points per win and 1 point per draw. Bolzano prefers the football way: 3 points per win and 1 point per draw. But Cardano doesn't like either way and has his own system to declare a winner. We define the graph of the tournament as the graph where each node represents a player and an edge goes from player v to player u if player v won at least one game against player u. Then Cardano states that a player v wins the tournament only if in the graph of the tournament there is a path from v to everyone else and there is none from any other player to v.

Recently, there has been a chess tournament of n players, numbered from 0 to $n-1$. The only information we have is the number of points of each player according to Abel's and Bolzano's criteria. Each player might have played any number of times with any other player. Determine if it is possible that player 0 won the tournament according to Cardano's criteria.

Input

The first line contains one integer t, the number of test cases ($1 \le t \le 10^4$). Each test case consists of three lines:

The first line of contains one integer n ($1 \le n \le 10^6$), the number of participants in the tournament.

The second line contains n integers $a_0, a_1, \ldots, a_{n-1}$ ($0 \le a_i \le 10^9$), where a_i is the number of points player i has obtained according to Abel's criteria.

The third line contains n integers $b_0, b_1, \ldots, b_{n-1}$ ($0 \le b_i \le 10^9$), where b_i is the number of points player i has obtained according to Bolzano's criteria.

The sum of n for all test cases won't exceed 10^6.

It is guaranteed that the given scoring corresponds to a valid tournament.

Output

For each test case, print a line with the word "`YES`" if it is possible that player 0 won the tournament according to Cardano's criteria. Otherwise, print a line with the word "`NO`".

Example

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

Problem J. Joyful Numbers

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

We say that an integer $n \geq 1$ is *joyful* if, by concatenating the digits 25 to the right of n, we get a perfect square. For example, 2 is a joyful number (as $225 = 15^2$) but 3 is not (as 325 is not a perfect square).

Given an integer k such that $1 \leq k \leq 10^9$, count the number of distinct prime factors of the k-th joyful number.

Input

The first line contains one integer t, the number of test cases ($1 \leq t \leq 4 \cdot 10^3$).

Each test case is given on a separate line containing an integer k ($1 \leq k \leq 10^9$).

Output

For each test case, print a line with a single integer: the number of distinct prime factors of the k-th joyful number.

Examples

standard input	standard output
2 1 4	1 2
1 1000000000	7

Note

The first joyful number is 2, which has one distinct prime factor. The fourth joyful number is $20 = 2 \cdot 2 \cdot 5$, which has two distinct prime factors.

Problem K. Königsberg Bridges

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

Given a graph, we say it is *Königsbergsy* if there is a simple path that goes through all of its bridges. Here, a *bridge* is an edge that disconnects the graph when removed. And recall that a simple path is a path that visits each vertex at most once.

Given a graph G, we want to add some edges to it to make it Königsbergsy. (You may add more than one edge between the same pair of vertices). Determine the maximum number of bridges that the resulting graph can have.

Input

The first line contains two integers n and m ($2 \le n \le 10^6$; $0 \le m \le 10^6$), the number of vertices and the number of edges of G.

Each of the next m lines contains two integers u_i, v_i ($0 \le u_i, v_i \le n - 1$), describing an edge between vertices u_i and v_i.

Output

Output one integer, the maximum number of bridges the resulting graph can have.

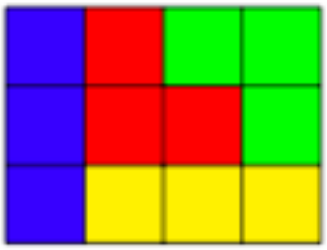
Examples

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

Problem L. Long Grid Covering

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

We have a grid of height 3 and width n, as well as pieces that occupy 3 adjacent cells. Given n, determine the number of ways to fill the grid so that each cell is covered by exactly one piece and no piece sticks out of the grid. Here there is an example of a way to fill a grid of width 4:



Notice that any piece will be a rotation of one of the pieces of this example. Find the answers modulo $10^9 + 7$.

Input

The first line contains an integer t, the number of test cases ($1 \le t \le 100$).

Each test case is given on a separate line containing an integer n ($1 \le n \le 10^{18}$), the width of the grid.

Output

For each test case, print a line with a single integer: the number of ways to fill the grid with aforementioned conditions modulo $10^9 + 7$.

Example

standard input	standard output
3 1 2 3	1 3 10