2021 China Collegiate Programming Contest
Qualification Round (Online)

Problem	Title
A / 1001	Median Problem
B / 1002	Kanade Doesn't Want to Learn CG
C / 1003	GCD on Tree
D / 1004	Primality Test
E / 1005	Monopoly
F / 1006	Nuh Heh Heh Aaaaaaaaaa
G / 1007	Occupying Grids
H / 1008	Subpermutation
I / 1009	Public Transport System
J / 1010	Bigraph Extension
K / 1011	Jumping Monkey
L / 1012	Contest Remake

Caution

1. Please refer to the online judge for the time limit.
2. Complication Error will be counted in penalty time.
3. Don't print any extra spaces at the end of each line. Do print newline ('\n' in C++) at the end of each line. Or you will get Presentation Error verdict.

Good luck and have fun!

Problem A. Median Problem

Input file:	standard input
Output file:	standard output
Memory limit:	256 megabytes

We consider an integer x as the *median* of a set A containing n distinct integers if it meets the following conditions:

- $x \in A$
- There are at least $\lfloor \frac{n+1}{2} \rfloor$ integers greater than or equal to x in set A.
- There are at least $\lfloor \frac{n+1}{2} \rfloor$ integers less than or equal to x in set A.

$\lfloor y \rfloor$ means the largest integer that does not exceed y. For example, if $A = \{1, 3, 4, 5, 7, 9\}$, then either 4 or 5 is the median of set A.

Given a tree with n nodes rooted at node 1. Each node u has an associated value a_u, where $a_1, a_2, \ldots, a_n$ is a permutation of n (every integer from 1 to n occurs exactly once). Each node u has another value b_u satisfying the following condition:

- b_u is the *median* of the set $\{a_u\} \cup \{b_v \mid \text{node } u \text{ is the parent of node } v\}$. (The parent of node v is the node directly connected to v on the path to the root.)

Unfortunately, you forget some of a_i and all b_i except b_1. Now you are wondering how many different permutations $a_1, a_2, \ldots, a_n$ that can match the a_i and b_1 you remember when the tree satisfies the condition above. We consider two permutations $p_1, p_2, \ldots, p_n$ and $q_1, q_2, \ldots, q_n$ different if there exists an index i satisfying $p_i \neq q_i$.

You are required to calculate the number of different permutations $a_1, a_2, \ldots, a_n$ modulo 998244353, for each $b_1 \in [1, n]$ respectively.

Input

The first line of the input contains an integer T ($1 \leq T \leq 80$), representing the number of test cases.

The first line of each test case contains an integer n ($2 \leq n \leq 80$), representing the number of nodes.

The second line of each test case contains $n - 1$ integers $p_2, p_3, \ldots, p_n$ ($1 \leq p_i < i$), where p_i represents the parent of node i.

The third line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i \leq n$), representing the value of each node. $a_i = 0$ means you forget a_i. It is guaranteed that all non-zero a_i are distinct.

It is guaranteed that there are at most 5 test cases satisfying $n > 10$.

Output

For each test case, output a single line containing n integers, the k-th of which is the answer when $b_1 = k$.

Don't print any extra spaces at the end of each line.

Example

standard input	standard output
2 4 1 1 1 0 0 0 2 5 1 1 2 2 0 0 1 5 0	0 6 6 0 0 2 4 0 0

Problem B. Kanade Doesn't Want to Learn CG

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	256 megabytes

Computational geometry and computer graphics are such hard parts of computer science that Kanade can't manage them. So she doesn't want to learn CG anymore. She goes to play basketball instead.

The only essential equipment in a basketball game is the ball and the hoop — a flat, rectangular backboard with a basket. We describe the hoop with a side view. Ignoring the thickness, the backboard is considered as a segment parallel to the y-axis, and the basket is considered as a segment parallel to the x-axis. The right end of the basket is connected to the backboard.

To simplify the model, we consider a basketball as a mass point. Taking only gravity into consideration, if we ignore the basket and the backboard, the trajectory of basketball will be a parabola $y = ax^2 + bx + c$ with $a < 0$. But the basketball is likely to hit the backboard, resulting in a change in trajectory. We consider the collision between a basketball and the backboard (including the endpoints) as a perfectly elastic collision, which means the velocity on the x-axis of the basketball will be reversed, and the velocity on the y-axis will remain the same. We ignore the court floor in this problem.

If the basketball passes through the basket (excluding the endpoints) from top to bottom, we consider the shoot is a goal. Once the basketball touches either of the endpoints of the basket, which means it hits the rim, the basketball will be bounced away and cannot make a goal. In addition, according to the rule, a basketball cannot pass through the basket from bottom to top, or it is a violation and cannot be counted as a goal.

Kanade knows the value of a, b, c and the position of the backboard and basket. She would like to know whether the shoot will be a goal if the basketball starts from $x = -114514^{1919810}$ and moves in the positive direction of the x-axis.

Input

The first line of input contains one integer T ($1 \le T \le 500$), indicating the number of test cases.

For each test case, the first line contains three integers a, b, c ($a < 0$), indicating the parameters of the parabola.

The second line of each test case contains five integers x_0, x_1, y_0, y_1, y_2 ($x_0 < x_1, y_1 < y_0 < y_2$), indicating that the two endpoints of the basket are (x_0, y_0) and (x_1, y_0), and the two endpoints of the backboard are (x_1, y_1) and (x_1, y_2).

It is guaranteed that the absolute value of all integers in the input won't exceed 10^4.

Output

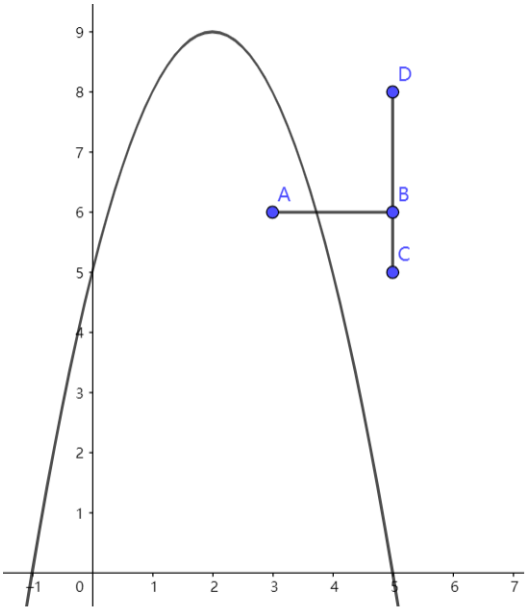
For each test case, if the shoot is a goal, output `Yes` in a single line, otherwise output `No` in a single line.

Example

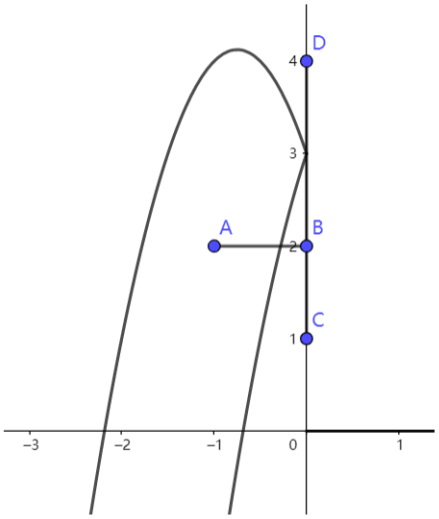
standard input	standard output
4 -1 4 5 3 5 6 5 8 -2 -3 3 -1 0 2 1 4 -1 -9 19 8 10 6 5 8 -1 9 19 8 10 4 3 6	Yes Yes No No

Note

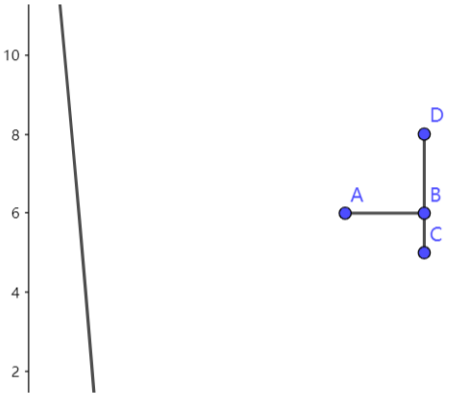
The samples are shown as follows. Segment AB represents the basket, and segment CD represents the backboard.



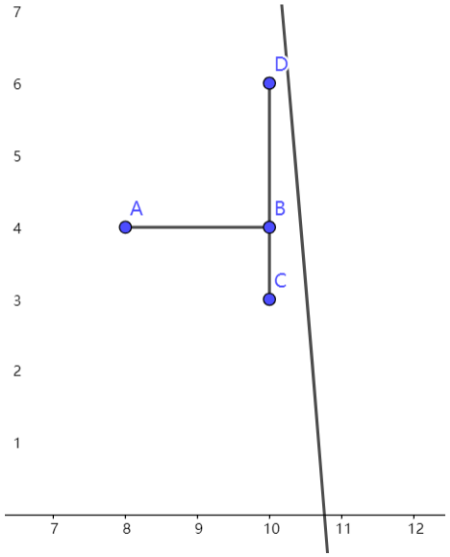
(I)



(II)



(III)



(IV)

Problem C. GCD on Tree

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	512 megabytes

You are given a tree with n nodes and $n-1$ edges that make all nodes connected. Each node i is assigned a weight a_i.

Now you need to perform m operations of two different types. One is to change the weight of a node. The other is to query the number of pairs (x, y) $(1 \le x \le y \le n)$ in the entire tree such that the greatest common divisor of the weights of all nodes on the shortest path from node x to node y is k.

Input

The first line contains a single integer T $(1 \le T \le 8)$, representing the number of test cases.

For each test case, the first line contains two integers n $(1 \le n \le 10^4)$ and m $(1 \le m \le 10^4)$, representing the number of nodes and operations respectively.

The second line contains n integers. The i-th integer a_i $(1 \le a_i \le 10^4)$ represents the initial weight of node i.

The third line contains $n-1$ integers. The i-th integer p_{i+1} $(1 \le p_{i+1} \le i)$ represents that there's an edge between node $i+1$ and node p_{i+1}.

The following m lines describe the operations.

- `0 u c`: change the weight of node u $(1 \le u \le n)$ to c $(1 \le c \le 10^4)$;
- `1 k`: query the number of pairs (x, y) $(1 \le x \le y \le n)$ in the entire tree such that the greatest common divisor of the weights of all nodes on the shortest path from node x to node y is k $(1 \le k \le 10^4)$.

Output

For each query, output the answer in a single line.

Example

standard input	standard output
2 8 6 2 1 5 6 7 2 3 4 1 2 2 4 4 1 7 1 2 0 2 2 1 2 1 5 0 7 4 1 2 3 5 1 2 3 1 1 1 1 0 1 6 1 2 1 3 1 6	3 9 1 17 4 2 2 1

Problem D. Primality Test

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	256 megabytes

A positive integer is called a *prime* if it is greater than 1 and cannot be written as the product of two smaller positive integers. A *primality test* is an algorithm for determining whether an input number is a prime. For example, the Miller–Rabin primality test is a probabilistic primality test. This problem is precisely the one about the primality test.

Let's define the function $f(x)$ as the smallest prime which is strictly larger than x. For example, $f(1) = 2$, $f(2) = 3$, and $f(3) = f(4) = 5$. And we use $\lfloor x \rfloor$ to indicate the largest integer that does not exceed x.

Now given x, please determine whether $g(x)$ is a prime.

$$g(x) = \left\lfloor \frac{f(x) + f(f(x))}{2} \right\rfloor$$

Input

The first line of the input contains an integer T ($1 \le T \le 10^5$), indicating the number of test cases.

Each test case contains an integer x ($1 \le x \le 10^{18}$) in a single line.

Output

For each test case, if $g(x)$ is a prime, output `YES` in a single line. Otherwise, output `NO` in a single line.

Example

standard input	standard output
2 1 2	YES NO

Note

When $x = 1$, $f(x) = 2$, $f(f(x)) = f(2) = 3$, then $g(x) = \left\lfloor \frac{2+3}{2} \right\rfloor = 2$, which is a prime. So the output is `YES`.

When $x = 2$, $f(x) = 3$, $f(f(x)) = f(3) = 5$, then $g(x) = \left\lfloor \frac{3+5}{2} \right\rfloor = 4$, which is not a prime. So the output is `NO`.

Problem E. Monopoly

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	256 megabytes

Little Rabbit loves playing a tabletop game called *Monopoly*. In the game, players roll the dice to move around the game board, buying and trading properties, and developing them with houses and hotels. Players collect rent from their opponents, with the goal being to drive them into bankruptcy.

One day, Little Rabbit invites Little Horse to play the game with him, but Little Horse stands him up. As a result, Little Rabbit can only play the game on his own. After a while, Little Rabbit finds out another way to play the game.

Let's consider the map of the game as n integers $a_1, a_2, \ldots, a_n$ placed in a circle. If the player lands on a_i, his score will change by a_i points. Specifically, if $a_i > 0$, his score will increase by $|a_i|$ points. If $a_i < 0$, his score will decrease by $|a_i|$ points. And of course, if $a_i = 0$, his score won't change.

As there is nobody else, Little Rabbit doesn't feel like rolling a dice. So he just moves on the map step by step. Formally, if he now lands on a_i and $i < n$, he will move to a_{i+1} in the next step. If he now lands on a_n, he will move to a_1 in the next step.

Initially, Little Rabbit has 0 points, and his first step is to land on a_1. Little Rabbit wonders how many steps he should take at least to reach **exactly** x points. (Specifically, since his initial score is 0, he needs 0 steps to reach 0 points.)

Input

The first line of the input contains an integer T ($1 \le T \le 20$), indicating the number of test cases.

The first line of each test case contains two integers n and m ($1 \le n, m \le 10^5$), indicating the number of integers on the map and the number of queries.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_1, a_2, \ldots, a_n \le 10^9$), indicating the integers on the map.

In the next m lines, each line contains an integer x ($-10^{12} \le x \le 10^{12}$), indicating a query.

It's guaranteed that $\sum n \le 5 \times 10^5$ and $\sum m \le 5 \times 10^5$.

Output

For each query, output an integer in a single line, indicating the number of steps he should take at least to reach **exactly** x points. If it is impossible to reach exactly x points, output -1.

Example

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

Problem F. Nun Heh Heh Aaaaaaaaaaa

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	256 megabytes

Vasily Tadokorov is a stringologist. He thinks a string is fragrant if it can be divided into two parts — `nunhehheh` as the prefix and a number of (excluding 0) `a` as the suffix. For example, `nunhehhehaaaaaa` is fragrant, but `nunhehheh` and `nunhehhehoooaaa` are not fragrant.

Today Vasily Tadokorov has some strings consisting of lowercase English letters. For each string, he wants to know how many subsequences of this string are fragrant. A string a is a subsequence of a string b if a can be obtained from b by deletion of several (including 0) characters.

Input

The first line contains an integer T ($1 \le T \le 1000$), denoting the number of strings.

Each of the next T lines contains a string S ($1 \le |S| \le 10^5$) consisting of lowercase English letters.

The total length of the strings in the input will not exceed 10^6.

Output

For each of the given T strings, output the answer modulo 998244353.

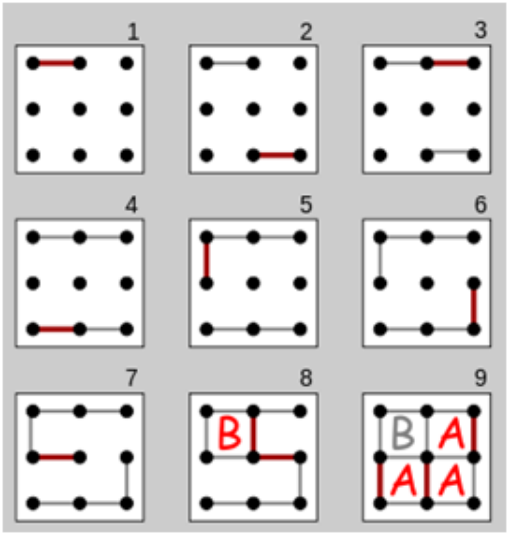
Example

standard input	standard output
2 nunhehhehahaahahahahahahaahaahahahahha nunhehhehhehhahaahahahaahaahahaaaahaa	114514 1919810

Problem G. Occupying Grids

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	256 megabytes

Little Rabbit and Little Horse are playing a game called *Occupying Grids*. The game is played on a square grid graph which has m dots each row and n dots each column. The players take turns to draw a segment that connects two adjacent dots. Once the four sides of a grid are all drawn, the player who draws the last side of the grid can occupy the grid and **immediately** get one more chance to draw another segment. (Sometimes two grids can be occupied at the same time, but the player also gets **only one** more chance to draw.) The game ends when no more segments can be drawn, and the player who occupies more grids will win the game. (Two players occupying the same number of grids results in a tie.) The picture below shows a game in a 3×3 square grid graph.



Little Rabbit and Little Horse both play the game for the first time. So in the beginning, they just draw the segments willy-nilly without any strategies. But after k steps, they are surprised to find that each grid has **at least 2** sides that are drawn. Little Rabbit wants to know whether he can win the game if both players play optimally from now on.

Little Rabbit cannot solve the problem, so he turns to you and shows you the k steps **in order**. But he forgets to note down which player each step belongs to, so you need to judge by yourself. The only thing you know is that the first step belongs to Little Rabbit.

Please note that the k steps given should strictly follow the rules. For example, if one player occupies a grid in some step, then the next step also belongs to him.

Input

The first line of the input contains an integer T ($1 \le T \le 50$), indicating the number of test cases.

The first line of each test case contains three integers n, m, k ($2 \le n, m \le 100$, $1 \le k \le 2nm - n - m$), indicating the number of rows and columns of the square grid graph, and the number of steps Little Rabbit shows you.

Then in the next k lines, each line contains three integers x, y, o ($1 \le x \le n$, $1 \le y \le m$, $0 \le o \le 1$) to describe a step. Here we use (x, y) to describe the point in the x-th row and the y-th column.

- `x y 0` ($1 \le x \le n$, $1 \le y < m$): a segment that connects (x, y) and $(x, y + 1)$.
- `x y 1` ($1 \le x < n$, $1 \le y \le m$): a segment that connects (x, y) and $(x + 1, y)$.

The first step belongs to Little Rabbit. It is guaranteed that after the k steps, each grid has at least 2 sides that are drawn. It is also guaranteed that any two steps are different.

Output

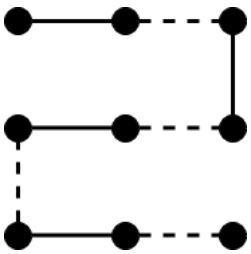
For the x-th test case, if Little Rabbit wins, output `Little Rabbit` in a single line. If Little Horse wins, output `Little Horse` in a single line. If the game ends in a tie, output `Tie` in a single line.

Example

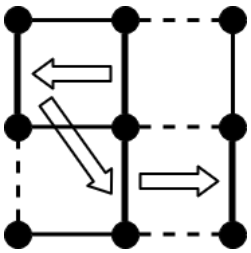
standard input	standard output
3 3 3 8 1 1 0 1 2 0 2 1 0 2 2 0 3 1 0 3 2 0 1 3 1 2 1 1 3 3 8 1 1 0 1 1 1 1 2 0 1 3 1 2 1 1 2 3 1 3 1 0 3 2 0 3 3 6 1 1 0 1 2 0 2 1 0 2 2 0 3 1 0 3 2 0	Little Rabbit Little Horse Tie

Note

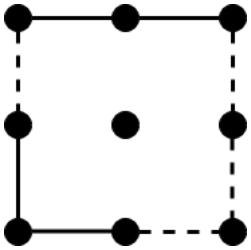
We use solid lines to indicate Little Rabbit's step and dashed lines to indicate Little Horse's step. The first test case of the sample input can be shown as follow.



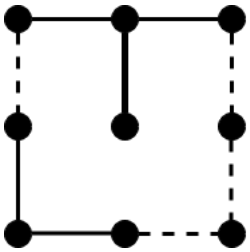
And now it is Little Rabbit's turn. It's obvious that Little Rabbit can occupy all the grids by the following steps.

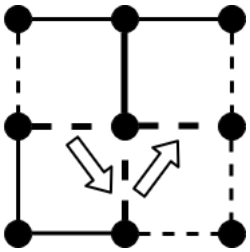


The second test case of the sample input can be shown as follow.



And now it is Little Rabbit's turn. He has no choice but to draw one of the four sides in the middle. Then Little Horse can occupy all the grids.





Problem H. Subpermutation

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	256 megabytes

A permutation of n is a sequence of length n in which each number from 1 to n appears exactly once. A *full-permutation* of n is a sequence that connects all permutations of n into one sequence in lexicographical order. Sequence $p_1, p_2, \ldots, p_n$ is lexicographically smaller than $q_1, q_2, \ldots, q_n$ if $p_i < q_i$ where i is the minimum index satisfying $p_i \neq q_i$.

Here are some symbols used in this problem:

Symbol	Definition	Example
p_n	the full-permutation of n	$p_3 = \{1,2,3,1,3,2,2,1,3,2,3,1,3,1,2,3,2,1\}$
S_n	the set of all permutations of n	$S_3 = \{\{1,2,3\},\{1,3,2\},\{2,1,3\},\{2,3,1\},\{3,1,2\},\{3,2,1\}\}$
$f(s,t)$	the number of contiguous subsequences in s that are equal to t	$f(\{1,2,12,1,2\},\{1,2\}) = 2$

Now given n and m, please calculate $\sum_{t \in S_m} f(p_n, t)$ modulo $10^9 + 7$.

Input

The first line contains one integer T ($1 \leq T \leq 10^5$), indicating the number of test cases.

The only line for each case contains two integers n ($1 \leq n \leq 10^6$) and m ($1 \leq m \leq n$), as described in the description.

Output

For each test case, output a single integer $\sum_{t \in S_m} f(p_n, t)$ modulo $10^9 + 7$.

Example

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

Note

For the third case in the sample, $p_3 = \{1,2,3,1,3,2,2,1,3,2,3,1,3,1,2,3,2,1\}$, $S_2 = \{\{1,2\},\{2,1\}\}$. There are 4 contiguous subsequences in p_3 that are equal to $\{1,2\}$ or $\{2,1\}$: $\{\underline{1,2},3,1,3,2,\underline{2,1},3,2,3,1,3,\underline{1,2},3,\underline{2,1}\}$.

Problem I. Public Transport System

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	256 megabytes

You are living in a country which has well-developed transportation. There are n cities numbered from 1 to n and m public transport routes numbered from 1 to m in the country. The route numbered i is a directed route from city u_i to city v_i, with a cost a_i and a preferential factor b_i.

To facilitate people's travel, the government has formulated a series of preferential measures. For a trip that starts from city s, passes through routes numbered $e_1, e_2, \ldots, e_k$ and ends in city t, the cost of each route is calculated as follows:

- For route e_1, the cost is a_{e_1}.
- For route e_i $(i > 1)$, if $a_{e_i} > a_{e_{i-1}}$, the cost is $a_{e_i} - b_{e_i}$, otherwise the cost is a_{e_i}.

The total cost of the trip is the sum of the costs of these routes.

You are now living in city 1. You want to find the minimum cost of traveling from city 1 to city k, for each $k \in [1, n]$ respectively.

Input

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

The first line of each test case contains two integers n, m $(2 \le n \le 10^5, 1 \le m \le 2 \times 10^5)$, representing the number of cities and routes.

For the following m lines, the i-th line contains four integers u_i, v_i, a_i, b_i $(1 \le u_i, v_i \le n,\ u_i \ne v_i,\ 1 \le b_i \le a_i \le 10^9)$, representing the route numbered i.

It is guaranteed that the sum of n over all test cases does not exceed 6×10^5 and the sum of m does not exceed 1.2×10^6.

Output

For each test case, output a single line containing n integers separated by a space, the k-th of which is the minimum cost of traveling from city 1 to city k, or -1 if you can't reach city k.

Don't print any extra spaces at the end of each line.

Example

standard input	standard output
2 4 4 1 2 3 2 2 3 4 1 1 3 7 5 4 3 2 1 4 8 4 2 3 3 1 3 6 3 4 2 10 5 1 2 8 2 3 2 4 3 4 2 7 7 3 4 4 2 1 2 8 1	0 3 6 -1 0 8 6 10

Problem J. Bigraph Extension

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	256 megabytes

There are $2n$ vertices (n is **even**), n of which belong to set A, and the rest n belong to set B. Initially, there are m undirected edges in the graph, where the two vertices of each edge are not in the same set. In addition, there is no common vertex between any two edges.

You are required to do the following operation multiple times:

- Choose a vertex from set A and another from set B on the condition that these two vertices are not directly connected by an edge. Then add an edge to connect the two vertices.

After that, when you choose any vertex in set A and any vertex in set B, the following conditions must be satisfied:

- The two vertices are connected, which means there exists a path that leads from one vertex to the other.
- The number of edges in the longest simple path between the two vertices is **strictly greater** than n. (In a simple path, each vertex is visited no more than once.)

Please **minimize** the number of edges you add.

Input

The first line contains a single integer T ($1 \le T \le 10^3$), representing the number of test cases.

For each test case, the first line of the input contains two integers n and m ($2 \le n \le 10^3, 0 \le m \le n$, n is **even**), representing the number of vertices in one set and the number of initial edges.

In the following m lines, each line contains two integers u and v ($1 \le u, v \le n$), indicating an edge between the u-th vertex in set A and the v-th vertex in set B. It's guaranteed that there is no common vertex between any two edges.

Output

For each test case, if there is no solution, print `-1` in a single line.

Otherwise, the first line of the output contains an integer k, indicating the minimum number of edges you add. In the following k lines, the i-th line contains two integers c_i and d_i ($1 \le c_i, d_i \le n$), indicating an edge you add between the c_i-th vertex in set A and the d_i-th vertex in set B.

As there may be multiple valid solutions, you need to output the answer which makes the sequence $c_1, d_1, c_2, d_2, \ldots, c_k, d_k$ have the smallest lexicographical order. Sequence $p_1, p_2, \ldots, p_n$ is lexicographically smaller than $q_1, q_2, \ldots, q_n$ if $p_i < q_i$ where i is the minimum index satisfying $p_i \neq q_i$.

Example

standard input	standard output
1 2 1 1 2	3 1 1 2 1 2 2

Note

For the sample, we can prove that the minimal k is 3. Here, we use A_i to indicate the i-th vertex in set A and B_i to indicate the i-th vertex in set B.

The longest path from A_1 to B_1 is $A_1 \to B_2 \to A_2 \to B_1$.

The longest path from A_2 to B_1 is $A_2 \to B_2 \to A_1 \to B_1$.

The longest path from A_1 to B_2 is $A_1 \to B_1 \to A_2 \to B_2$.

The longest path from A_2 to B_2 is $A_2 \to B_1 \to A_1 \to B_2$.

The answer $(1, 1), (2, 2), (2, 1)$ is also valid, but it is not lexicographically smallest.

Problem K. Jumping Monkey

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	256 megabytes

There is a tree with n nodes and $n-1$ edges that make all nodes connected. Each node i has a **distinct** weight a_i. A monkey is jumping on the tree. In one jump, the monkey can jump from node u to node v if and only if a_v is the greatest of all nodes on the shortest path from node u to node v. The monkey will stop jumping once no more nodes can be reached.

For each integer $k \in [1, n]$, calculate the maximum number of nodes the monkey can reach if he starts from node k.

Input

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

The first line of each test case contains an integer n ($1 \le n \le 10^5$), representing the number of nodes.

Each of the following $n-1$ lines contains two integers u, v ($1 \le u, v \le n$), representing an edge connecting node u and node v. It is guaranteed that the input will form a tree.

The next line contains n **distinct** integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), representing the weight of each node.

It is guaranteed that the sum of n over all test cases does not exceed 8×10^5.

Output

For each test case, output n lines. The k-th line contains an integer representing the answer when the monkey starts from node k.

Example

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

Note

For the second case of the sample, if the monkey starts from node 1, he can reach at most 4 nodes in the order of $1 \to 3 \to 2 \to 4$.

Problem L. Contest Remake

Input file:	`standard input`
Output file:	`standard output`
Memory limit:	512 megabytes

The annual Chengdu Children Poker Contest (CCPC) has begun! Unfortunately, the cards have been all blown away during the contest and the contest cannot go ahead. You, as the contest organizer, need to prepare a rematch and reproduce a pack of cards for the rematch.

The cards used in the contest are quite special — each card represents a set of integers. For a pack of m cards, we denote the set represented by each card as $S_1, S_2, \ldots, S_m$. The pack of the cards should meet the following conditions:

- The elements in each set are unordered and distinct.
- For every i, $S_i \subset \mathbb{Z}^+$ (the set of all positive integers), and $S_i \neq \emptyset$.
- For every $1 \leq i < j \leq m$, $S_i \neq S_j$.
- For every i, $\prod_{x \in S_i} x \leq C$, where C is a given constant.

What's more, a mystery guy tells you the reason why the cards are blown away is that you use some ominous integers. There are n ominous integers, and you should avoid using them in case your cards get blown away again. We denote the set of ominous integers as T. That is:

- For every i, $S_i \cap T = \emptyset$.

You want to reproduce as many cards as possible. Now given C and the n ominous integers, please calculate the maximum number of cards in the pack.

Input

The first line contains a positive integer T ($1 \leq T \leq 20$), indicating the number of test cases.

For each test case, the first line contains two integers n, C ($0 \leq n \leq 10^5, 1 \leq C \leq 10^9$), indicating the number of ominous integers and the given constant.

The second line contains n positive integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq C$), indicating the ominous integers. It is guaranteed that all the n integers are distinct.

It is guaranteed that $\sum n \leq 7 \times 10^5$, and there are at most 10 test cases that meet $C > 10^6$.

Output

For each test case, print one integer in a single line, indicating the maximum number of cards in the pack.

Example

standard input	standard output
2 1 6 3 2 10 3 5	9 17