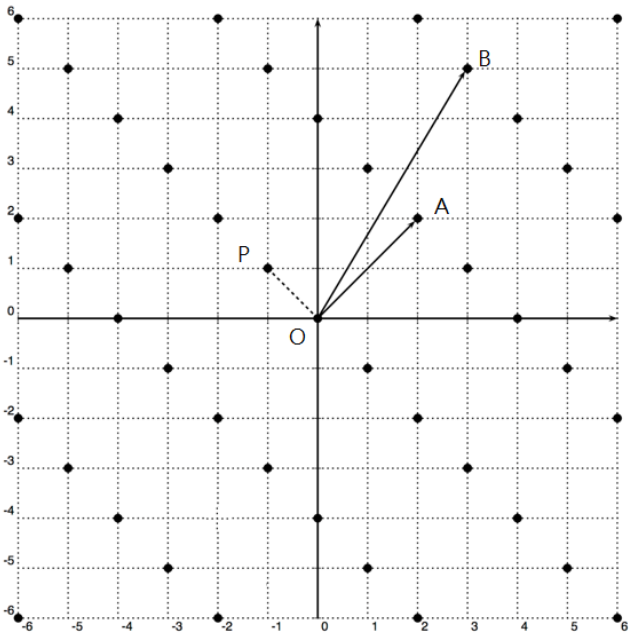
Problem 1. A Special Version of Closest Pair Problem

(Time Limit: 3 seconds)

Problem Description

The closest pair problem is a classic problem in computational geometry. The ordinary version is given n points in a 2D plane, to find a pair of points with shortest distance between them. Here, we consider a variant of the closest pair problem. In this problem, the points are generated by all integer combinations of $\vec{A} = (x_A, y_A)$ and $\vec{B} = (x_B, y_B)$, i.e., the set of all points are $\{\alpha\vec{A} + \beta\vec{B}: \alpha, \beta \in Z\}$ where Z is the set of all integers. For example, the figure below shows an instance of this problem: the points are generated by $\vec{A} = (2,2)$ and $\vec{B} = (3,5)$. The closest pair is $\{O, P\}$, and the shortest distance is $\sqrt{2}$.



The algorithm for the ordinary version does not work, since there are infinitely many points now. Please write a program to compute the distance between the closest pair.

Input Format

The first line of the input contains an integer $T \leq 20$ which is the number of test cases. Each test case contains 4 integers x_A, y_A, x_B, y_B separated by blanks where $\vec{A} = (x_A, y_A)$ and $\vec{B} = (x_B, y_B)$. Note: all integers in the input are in $[-50000, 50000]$.

Output Format

The output contains one line for each test case. Output the distance between the closest pair. Note: You should print the value to 2 decimal places.

Example

Sample Input:	Sample Output:
3 2 2 3 5 100 1001 123 1234 399 5000 4999 50000	1.41 14.21 1009.00

Problem 2. Healthy Network

(Time Limit: 1 second)

Problem Description

A human body is healthy if all the functional proteins are regularly produced. The underlying production process of proteins is a complicated network, because all the proteins affect the production of the others. Thus, protein-protein interaction network represents the interactions among all proteins in our human body. For each pair of proteins, the up-stream protein may increase or reduce the production rate of down-stream proteins, which is represented by a positive or negative integer, respectively. For a consecutive chain of interactions of proteins, the production rate of last protein is thus the summation of all rates along the path. If the summation of production of any protein in the network is negative through some cyclic chain reactions, the network is considered to be at risk of diseases (see left figure below), because the production rate of this protein will be continuously decreased over time. On the other hand, if all proteins are free of such negative cyclic chain reactions, it is considered healthy. Note that there is at most one interaction between each pair of proteins and no self-interaction of the same protein.

Given a protein-protein interaction network, you are asked to write a program to determine if the network is healthy or not.

Technical Specification

- A positive integer n, $3 \leq n \leq 300$, representing the number of proteins, and the protein IDs range from 1 to n.
- A positive integer m, $3 \leq m \leq 5000$, representing the number of paired interactions.
- The production of each interaction is an integer ranging from -600 to 600.
- The number of test cases is at most 30.

Input Format

The first line of the input file contains an integer indicating the number of test cases. Each test case starts with a line containing the number of proteins n and number of interactions m, separated by a white space. Each of the following m lines contains a triple of integers (a, b, c) separated by a white space. Each triple represents protein *a* contributes c production to protein b.

Output Format

For each test case, output "Yes" in one line if the network is healthy and "No" if it is at risk of disease.

Example

Sample Input:	**Sample Output:**
2 3 3 1 2 1 2 3 -3 3 1 1 4 4 1 2 3 2 3 -3 3 1 1 1 4 -2	No Yes

Problem 3. Tree Analyses

(Time Limit: 3 seconds)

Problem Description

It is often that a tree is used to represent the hierarchical structure of a set of objects. In this problem, all trees are rooted trees. For any two vertices u, v on a tree, the distance between u and v, denoted by $d(u,v)$, is defined to be the number of edges on the path from u to v. For example, in Figure 1, $d(7,5)=2$ and $d(1,2)=3$. For each vertex v on a tree, we define $h(v)$ to be its *height*, which is the maximum distance between v and any leaf in the subtree rooted at v; and we define $r(v)$ to be its *eccentricity*, which is the maximum distance between v and any vertex on the tree. For example, in Figure 1 the height of vertex 6 is 0, the height of vertex 2 is 3, and the height of vertex 4 is 5. In this example, since the vertex furthest from vertex 7 is vertex 9 and the vertex furthest from vertex 5 is vertex 6, we have $r(7)=d(7,9)=4$ and $r(5)=d(5,6)=5$. For a tree T, define $H^*(T)$ to be the total height of all vertices and define $R^*(T)$ to be the total eccentricity of all vertices. That is, $H^*(T)=\sum_{v\in V} h(v)$ and $R^*(T)=\sum_{v\in V} r(v)$, where V is the vertex set of T.

Please write a program to compute $H^*(T)$ and $R^*(T)$ for a given rooted tree T.

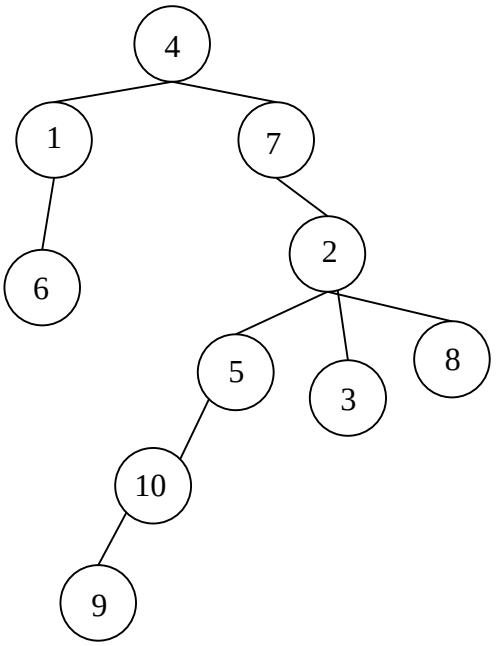


Figure 1. A rooted tree.

Technical Specification

- There are at most 10 test cases.
- The number of vertices, n, is an integer between 1 and 10^6.

Input Format

The first line is an integer t, $1 \leq t \leq 10$, indicating the number of test cases. Each test case starts with one line containing an integer n, $1 \leq n \leq 10^6$, indicating that there are n vertices labeled by $1, 2, \ldots, n$. Then, $n-1$ lines follow, each of which contains two integers p and c, $1 \leq p, c \leq n$, indicating there is an edge between vertex p and vertex c, where p is the parent of c.

Output Format

For each test case, print a line containing two integers indicating the value of $H^*(T)$ and $R^*(T)$.

Example

Sample Input:	Sample Output:
2 10 4 1 1 6 4 7 7 2 2 5 2 3 2 8 5 10 10 9 9 2 9 1 7 1 8 5 6 2 4 1 3 5 1 5 2	16 54 4 31

Problem 4. Fair points

(Time Limit: 1 second)

Problem Description

Given some points in the plane, find a point p among the input such that the number of points on each side of any line going through p is at most $n/2$.

Technical Specification

- The number of test cases is between 5 and 10. For each test case, there are n points, an "odd" integer between 1 and 10^5. For each point, the coordinates (x, y) satisfies $-10^6 < x < 10^6$ and $-10^6 < y < 10^6$.

Input Format

The first line is an integer indicating the number of test cases. Each test case consists of $n + 1$ lines, where the first line is the integer n, which is the number of given points. Each of the following n lines consists of 2 integers, which is the coordinates of a point. Consecutive integers in a line are separated by a space.

Output Format

For each test case, output the coordinate (two number separated a space) of the requested point if it exists, and 0 otherwise. If there is more than one solution, output the one that is lexicographically smallest.

Example

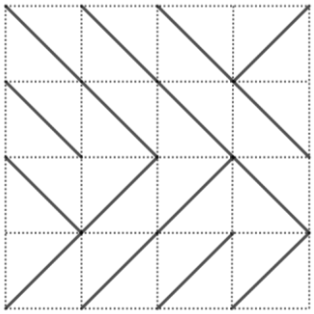
Sample Input:	Sample Output:
2 3 1 0 5 0 2 3 5 -2 3 1 4 4 0 0 2 -3 -4	0 0 2

Problem 5. Problem Mirrors

(Time Limit: 6 seconds)

Problem Description

There is a strange optic laboratory in Yen's school. The floor of the laboratory is a grid of m rows and n columns, and each grid cell is a square of $1\,m^2$. In each grid cell, there is a double-sided mirror of length $\sqrt{2}$ meters placed on one of the diagonals. For example, the laboratory of a 4-by-4 grid floor could be as follows.



Yen is testing a laser pointer. It can be placed on the north of the laboratory, and its direction is to the south. Yen will use the laser pointer to caste a ray toward the center of a grid of some column. Then he observes whether the ray will exit the laboratory from a desired column the south side. If not, he will adjust some mirror. For examples, Yen castes the ray toward the center of grid on column 1, then he observes that the ray exits from column 1 of the south side like Figure 1 (a). He probably will flip the mirror in the grid on the intersection of row 4 and column 2. Then the ray exits from column 3 of the south side like Figure 1 (b).

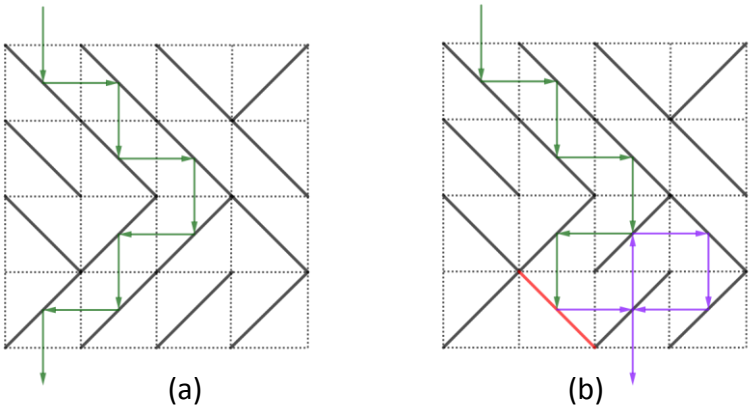


Figure 1

After a few adjustments, Yen is tired. He asks you to write a program to simulate the testing. There are only two operations.

1. $x\ y$: Flip the mirror in the grid on row x and column y.
2. z: From north to south, caste a ray toward the center of the grid on column z. Your program should output -1 if the ray does not exit from the south side. Otherwise, output the column of the gird where the ray exits.

Technical Specification

- T, the number of test cases, is at most 20.
- $n\ m$, the total number of grid cells, is at most 10^5.
- q, the number of operations, is no more than 10^5.

Input Format

The first line of the input file contains an integer T indicating the number of test cases. Each test case consists of 3 parts. The first part is a line containing 3 positive integers n, m, q separated by blanks where n, m, q are the numbers of rows, columns and operations, respectively. The second part consists of n lines. The i-th of them describes the mirror configuration of the grid cells on row i. If its j-th character is '\', the mirror in the grid cell on column j is placed from northeast to southwest. If its j-th character is '/', the mirror in the grid cell on column j is placed from northwest to southeast. There are q lines in the third part. Each of them is either $1\ x\ y$ (flip the mirror in the grid on row x and column y) or $2\ z$ (from north to south, caste a ray toward the center of the grid on column z)

Output Format

For each operation of the second type, output -1 if the ray does not exit from the south side. Otherwise, output the column of the gird where the ray exits.

Example

Sample Input:	Sample Output:
2 1 1 2 / 1 1 1 2 1 4 4 7 \\\/ \\\\ \//\ //// 2 1 1 4 2 2 1 2 2 1 3 4 1 4 4 2 1	-1 1 3 -1 -1