A – YAPTCHA

The math department has been having problems lately. Due to immense amount of unsolicited automated programs which were crawling across their pages, they decided to put Yet-Another-Public-Turing-Test-to-Tell-Computers-and-Humans-Apart on their webpages. In short, to get access to their scientific papers, one have to prove yourself eligible and worthy, i.e. solve a mathematic riddle.

However, the test turned out difficult for some math PhD students and even for some professors. Therefore, the math department wants to write a helper program which solves this task (it is not irrational, as they are going to make money on selling the program).

The task that is presented to anyone visiting the start page of the math department is as follows: given a natural n, compute

$$S_n = \sum_{k=1}^{n} \left[\frac{(3k+6)!+1}{3k+7} - \left[\frac{(3k+6)!}{3k+7} \right] \right]$$

where $[x]$ denotes the largest integer not greater than x.

Input

The first line contains the number of queries t ($t \leq 10^6$). Each query consist of one natural number n ($1 \leq n \leq 10^6$).

Output

For each n given in the input output the value of S_n.

Example

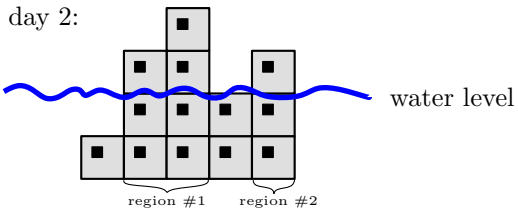
Input	Output
13 1 2 3 4 5 6 7 8 9 10 100 1000 10000	0 1 1 2 2 2 2 3 3 4 28 207 1609

B – Skyscrapers

In a seaside village, there is an avenue of skyscrapers. Each skyscrapers is 100m wide and has certain height. Due to very high price of parcels, any two consecutive skyscrapers are adjacent. The avenue lies close to the beach so the street is exactly at the sea level.

Unfortunately, this year, due to the global warming, the sea level started to increase by one meter each day. If the skyscraper height is no greater than the current sea level, it is considered flooded. A *region* is a maximal set of non-flooded, adjacent skyscrapers. This term is of particular importance, as it is sufficient to deliver goods (like current, carrots or cabbages) to any single skyscraper in each region. Hence, the city major wants to know how many regions there will be in the hard days that come.

An example of an avenue with 5 skyscrapers after 2 days is given below.



Input

The input contains several test cases. The first line contains an integer t ($t \leq 15$) denoting the number of test cases. Then t test cases follow. Each of them begins with a line containing two numbers n and d ($1 \leq n, d \leq 10^6$), n is the number of skyscrapers and d is the number of days which the major wants to query. Skyscrapers are numbered from left to right. The next line contains n integers $h_1, h_2, \ldots, h_n$ where $1 \leq h_i \leq 10^9$ is the height of skyscraper i. The third line of a single test case contains d numbers t_j such that $0 \leq t_1 < t_2 < \ldots < t_{d-1} < t_d \leq 10^9$.

Output

For each test case output d numbers $r_1, r_2, \ldots, r_d$, where r_j is the number of regions on day t_j.

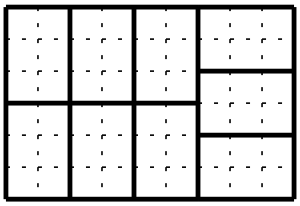
Example

Input	Output
2 3 3 1 2 3 1 2 3 5 3 1 3 5 1 3 0 2 4	1 1 0 1 2 1

C – Business Cards

Running a paper shop is not an easy job, especially with harsh customers. Today they brought their own rectangular sheets of paper, asking you to cut it into rectangular business cards of specific size. Moreover, they require that all the paper (which may not be cheap, but is definitely not *that* expensive!) has to be used, i.e. no tiny bit may be left over. Moreover, the brilliant idea of cutting the sheet into very small pieces, and then gluing them together in desired sheets was laughed at.

An example of a 9×6 paper sheet divided into 2×3 cards is given below.



Input

The input contains several test cases. The first line contains the number of test cases t ($t \leq 10^5$). Then t test cases follow. Each of them consists of one line containing four integers a, b, c, d ($1 \leq a, b, c, d \leq 10^9$). Numbers a and b are dimensions of each business card; c and d are dimensions of the paper sheet.

Output

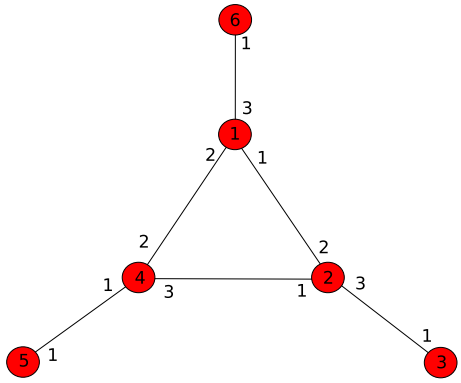
For each test case output one line containing word `YES` if it is possible to divide the whole sheet into business cards, and `NO` otherwise.

Example

Input	Output
4 2 3 9 6 2 3 8 6 2 3 6 8 2 3 5 7	YES YES YES NO

D – Museum

There is this big museum, full of fancy rooms and shiny corridors. It is so large that planning any tour in it becomes a serious issue. This is where your help is necessary. You are to help in planning signs that will make navigation through the whole building much easier. The idea is that if a room has d doors leading through corridors to other rooms, these doors and corresponding corridors are (locally) labeled with numbers $1, 2, \ldots, d$. Then all visitors are advised to follow a simple procedure. If they are in room v at the very beginning of their tour, they should choose door labelled with 1 and pass through the corresponding corridor If they are in room v and they entered it through door labelled with i, they should pick the door labelled with the next number (i.e. $i + 1$ if $i < d$ and 1 if $i = d$) and pass through corresponding corridor. Here is a simple example, in which tourists start in room 1 and visit rooms $1, 2, 3, 4, 5, 6$ in this order passing through each corridor at least once:



Exhibits in this museum are not only in rooms but also in corridors connecting different rooms. After all, the corridors are well suited for displaying paintings and photography! Thus we want to ensure that the tourists that follow the rules will pass through each corridor at least once, assuming they do not get bored easily and walk long enough, **irrespectively of the room they started the tour in**. Your task is to find such a labelling.

It turns out that there are at most 3 corridors outgoing from each room and the whole museum is connected, i.e. it is possible to walk between any two rooms, possibly passing through other rooms along the way. All corridors outgoing from a single room lead to different rooms.

Input

The input contains several test cases. The first line contains the number of test cases t ($t \leq 100$). Each test begins with a line containing the number of rooms n ($3 \leq n \leq 10^5$). The next n lines contain description of all corridors. i-th of them described corridors connecting the i-th room with others. It begins with an integer d ($1 \leq d \leq 3$), the number of doors in this room. d integers $r_1, r_2, \ldots, r_d$ follow, giving numbers of rooms that those doors lead to ($1 \leq r_j \leq n$ and $r_j \neq r_k$ if $j \neq k$ and $r_j \neq i$). All corridors are bidirectional, so if there is door from room x to room y, there is door from room y to room x as well. Total size of the input will not exceed 50MB.

Output

For each test case output exactly n lines. i-th of them should contain numbers of rooms directly connected by corridors with room i, in order of their assigned labels. You may assume that a valid labelling of doors always exists, you just need to find one.

Example

Input	Output
2 6 3 4 2 3 3 5 1 3 3 6 1 2 1 1 1 2 1 3 4 2 2 4 2 1 3 2 2 4 2 1 3	3 4 2 3 3 5 3 1 3 6 1 2 1 1 1 2 1 3 2 2 4 2 1 3 2 2 4 2 1 3

E – Morphing is fun

Morphic is a tree that grows very rapidly, bringing happiness to its owner. It has a single trunk consisting of a number of cells stacked one on top of another. Each cell has one of n possible colors which determine the way it mutates during the night, while nobody can see it. Florists denote these colors by the first n small letters of the English alphabet and know exactly into how many cells, and of what colors, a cell of each color divides. In fact, they have wrote their knowledge down simply with n nonempty words, each word representing the resulting sequence of colors.

A seed of a Morphic has a single cell of color a and is rooted firmly in the ground. As long as the Morphic is still alive, each night all its cells simultaniously morph according to the aforementioned rules, possibly causing an exponential growth because each new cell is of the same size as the original one. For example, if rules say that a becomes ab, and b becomes ca, then after two nights a seed will evolve to a trunk consisting of 4 cells: $abca$.

Therefore the top of a Morphic is usually hidden in clouds. The only way to tell if it is still alive is to check if visible part of the trunk is changing colors. In order to do so, one can build enormously high (yet still of constant height) tower, and watch from its top a fixed fragment of the trunk.

As you can easily see, it is either sufficient to observe first k cells from the bottom for some fixed k, or no matter how high the tower is, you will not be able to tell for sure if a Morphic died. The latter happens when for every k, rules cause the k-th cell to eventually stop changing colors, even though the tree is still alive and mutating.

To prevent waste of money on building such enormous towers, you are to write a program that determines if it is possible to monitor health of a Morphic.

Input

The input contains several Morphics descriptions. The first line contains the number of descriptions t ($t \leq 10^4$) that follow. Each of them begins with the number of colors n ($1 \leq n \leq 26$). Next n lines contain the rules by which the Morphic grows. The i-th one describes the sequence of colors in bottom-up order obtained from a single cell of i-th color. Each line contains at most 100 lowercase English letters.

Output

For each test case output one line containing YES if building of a tower is pointless (as in: YES, we can save money!). Otherwise output NO.

Example

Input	Output
4 2 ab a 3 ba c c 3 ba c b 3 bbbbbbbbbbbbbbb ccccccccccccc c	YES YES NO YES

F – Tower

Alan loves to construct the towers of building bricks. His towers consist of many cuboids with square base. All cuboids have the same height $h = 1$. Alan puts the consecutive cuboids one over another:

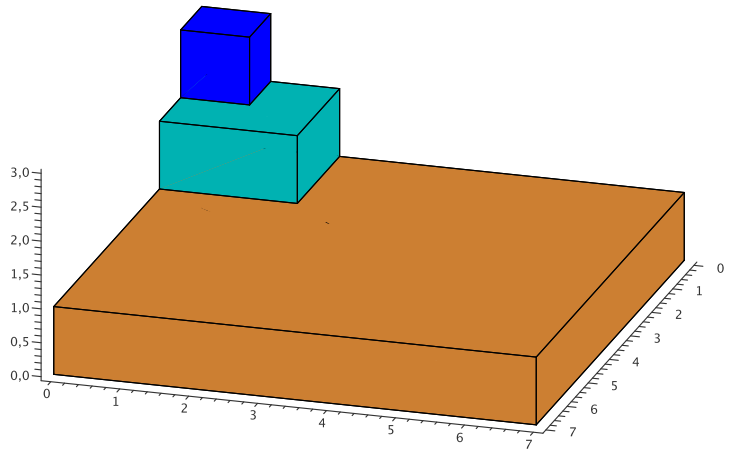


Figure 1: Tower of three bricks when Alan fixes $a_2 = 2$.

Recently in math class, the concept of volume was introduced to Alan. Consequently, he wants to compute the volume of his tower now. The lengths of cuboids bases (from top to bottom) are constructed by Alan in the following way:

1. Length a_1 of the first square is one.
2. Next, Alan fixes the length a_2 of the second square.
3. Next, Alan calculates the length a_n $(n > 2)$ by $2\,a_2\,a_{n-1} - a_{n-2}$. Do not ask why he chose such a formula; let us just say that he is a really peculiar young fellow.

For example, if Alan fixes $a_2 = 2$, then $a_3 = 8 - a_1 = 7$; see Figure 1. If Alan fixes $a_2 = 1$, then $a_n = 1$ holds for all $n \in \mathbb{N}$; see Figure 2.

Now Alan wonders if he can calculate the volume of tower of N consecutive building bricks. Help Alan and write the program that computes this volume. Since it can be quite large, it is enough to compute the answer modulo given natural number m.

Input

The input contains several test cases. The first line contains the number t ($t \leq 10^5$) denoting the number of test cases. Then t test cases follow. Each of them is given in a separate line containing three integers a_2, N, m ($1 \leq a_2, m \leq 10^9$, $2 \leq N \leq 10^9$) separated by a single space, where a_2 denotes the fixed length of second square in step 2, while N denotes the number of bricks constructed by Alan.

Output

For each test case (a_2, N, m) compute the volume of tower of N consecutive bricks constructed by Alan according to steps (1–3) and output its remainder modulo m.

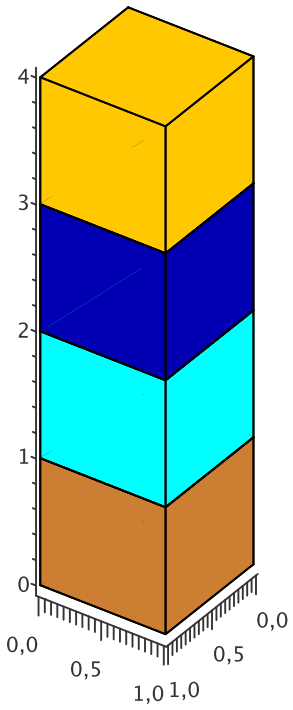


Figure 2: Tower of four bricks when Alan fixes $a_2 = 1$.

Example

Input	Output
3 2 3 100 1 4 1000 3 3 1000000000	54 4 299

G – Suffix reconstruction

Given a text $s[1..n]$ of length n, we create its suffix array by taking all its suffixes:

$$s[1..n], s[2..n], \ldots, s[n..n]$$

and sorting them lexicographically. As a result we get a sorted list of suffixes:

$$s[p(1)..n], s[p(2)..n], \ldots, s[p(n)..n]$$

and call the sequence $p(1), p(2), \ldots, p(n)$ the suffix array of $s[1..n]$. For example, if $s = abbaabab$, the sorted list of all suffixes becomes:

$$aabab, ab, abab, abbaabab, b, baabab, bab, bbaabab$$

and the suffix array is $4, 7, 5, 1, 8, 3, 6, 2$.

It turns out that it is possible to construct this array in a linear time. Your task will be completely different, though: given $p(1), p(2), p(3), \ldots, p(n)$ you should check if there exist at least one text consisting of lowercase letters of the English alphabet for which this sequence is the suffix array. If so, output any such text. Otherwise output -1.

Input

The input contains several descriptions of suffix arrays. The first line contains the number of descriptions t ($t \leq 100$). Each description begins with a line containing the length of both the text and the array n ($1 \leq n \leq 500000$). Next line contains integers $p(1), p(2), \ldots, p(n)$. You may assume that $1 \leq p(i) \leq n$ and no value of $p(i)$ occurs twice. Total size of the input will not exceed 50MB.

Output

For each test case output **any** text resulting in the given suffix array. In case there is no such text consisting of lowercase letters of the English alphabet, output -1.

Example

Input	Output
5 2 1 2 2 2 1 3 2 3 1 6 3 4 5 1 2 6 14 3 10 2 12 14 5 13 4 1 8 6 11 7 9 7 5 1 7 4 3 2 6	ab aa bab suffix reconstruction issofun

H – Two professors

There are two professors at the great Academy of X that really do not get along with each other. In order not to reveal their names, we will call them 1 and 2. The Academy employs exactly n professors; each of them has to give exactly one lecture. As their schedules are rather tight (they are professors, remember?), the starting and the ending time of each lecture is already fixed. However, it is not yet fixed where each lecture will take place. Obviously, it is impossible to schedule two lectures in the same room if their durations overlap; on the other hand, it is possible if one of them starts exactly at the same time that the other one ends. Your tasks is to find the minimal number of rooms allowing to arrange all the lectures. But know that professors 1 and 2 hate each other so much that they will never give their lectures in the same room.

Input

The input contains several test cases. The first line contains the number of test cases t ($t \leq 250$). Each test begins with a line containing the number of professors n ($2 \leq n \leq 10^5$). Next n lines follow, i-th of which contains two integers $start_i$ and end_i ($0 \leq start_i < end_i \leq 10^9$), the starting and the ending time of the lecture that the i-th professor gives, respectively. Total size of the input will not exceed 50MB.

Output

For each test case output the minimal number of rooms necessary to schedule all the lectures.

Example

Input	Output
4 2 0 10 10 20 3 0 10 10 20 10 20 5 4 14 3 13 2 12 1 11 0 10 4 0 10 10 20 20 30 30 40	2 2 5 2

I – Counting heaps

We are given a rooted tree of n vertices. The vertices are to be labeled with numbers $1, 2, \ldots, n$ so that each label is unique and the heap condition holds, i.e. the label of any vertex is less than the label of its parent. How many such labellings exist? Since this number may be quite large, calculate only its remainder modulo m.

Input

The input contains several tree descriptions. The first line contains the number of input trees t ($t \leq 250$). Each tree description begins with a line containing the size of the tree n ($1 \leq n \leq 500000$) and an integer m ($2 \leq m \leq 10^9$). $n-1$ lines follow, i-th of which contains $p(i+1)$, the number of the parent of the $i+1$-th vertex ($1 \leq p(i+1) \leq i$). Vertex number 1 will be the root in each tree, so its parent will not be given. Total size of the input will not exceed 50MB.

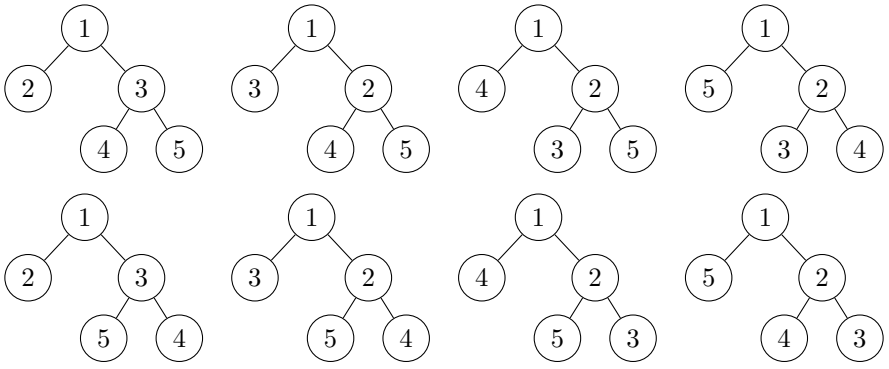
Output

For each tree output the number of its valid labellings modulo given m.

Example

Input	Output
4 3 1000000 1 1 4 1000000 1 1 1 5 1000000 1 2 3 4 5 1000000 1 1 3 3	2 6 1 8

The 8 possible labellings from the last example test case are as follows:



J – In case of failure

To help their clients deal with faulty Cash Machines, the board of The Planar Bank has decided to stick a label expressing sincere regret and sorrow of the bank about the failure on every ATM. The very same label would gently ask the customer to calmly head to the nearest Machine (that should hopefully work fine).

In order to do so, a list of two-dimensional locations of all n ATMs has been prepared, and your task is to find for each of them the one closest with respect to the Euclidean distance.

Input

The input contains several test cases. The very first line contains the number of cases t ($t \leq 15$) that follow. Each test cases begin with the number of Cash Machines n ($2 \leq n \leq 10^5$). Each of the next n lines contain the coordinates of one Cash Machine x, y ($0 \leq x, y \leq 10^9$) separated by a space. No two points in one test case will coincide.

Output

For each test case output n lines. i-th of them should contain the squared distance between the i-th ATM from the input and its nearest neighbour.

Example

Input	Output
2 10 17 41 0 34 24 19 8 28 14 12 45 5 27 31 41 11 42 45 36 27 15 0 0 1 2 2 3 3 2 4 0 8 4 7 4 6 3 6 1 8 0 11 0 12 2 13 1 14 2 15 0	200 100 149 100 149 52 97 52 360 97 5 2 2 2 5 1 1 2 4 5 5 2 2 2 5