Problem A. Kick Start

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Kick Start is a global online coding competition brought by Google. Participants can compete in online rounds held throughout the entire year, and will have the opportunity to develop and grow their programming abilities while getting a glimpse into the technical skills needed for a career at Google.

Mr. Panda is a Kick Start enthusiast who doesn't miss any Kick Start round of any year. He is now reading the 2019 schedule of rounds and wonders what's the date of the next Kick Start round (**excluding today**). Can you help Mr. Panda determine that date?

Any given date in this problem will be in the format of a month abbreviation followed by a date ordinal number. Recall that month abbreviations are 'Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sept', 'Oct', 'Nov', and 'Dec'; and ordinal numbers are '1st', '2nd', '3rd', '4th', '5th', etc.

Input

The first line of the input gives the number of test cases, T ($1 \le T \le 100$). T test cases follow.

For each test case, the first line contains an integer n ($1 \le n \le 20$), the number of scheduled Kick Start rounds for this year.

In the next n lines, each line contains a date in the year 2019, in the format of a month abbreviation followed by a date ordinal number. Note that scheduled dates may **not** be arranged in chronological order, and all scheduled dates are distinct.

The last line contains the date of today, also in the format of a month abbreviation followed by a date ordinal number.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from 1), and `y` is the date of the next Kick Start round or "See you next year" (quotes for clarity) if there is no following Kick Start round of this year.

Example

standard input	standard output
2 3 Jan 1st Feb 2nd Mar 3rd Jan 2nd 8 Mar 24th Apr 20th May 26th Jul 28th Aug 25th Sept 29th Oct 19th Nov 17th Nov 17th	Case #1: Feb 2nd Case #2: See you next year

Problem G. Game on the Tree

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	256 mebibytes

Mr. Panda and Mr. Sheep are playing a game on a tree with n vertices. Initially, there is a token on the vertex **1**. Mr. Panda and Mr. Sheep take turns and Mr.Panda moves first.

In each turn, a player must move the token to another vertex. There is a restriction that except for the first step, the distance the token moved must be strictly greater than the distance it moved in the previous turn by the opponent. If there is no valid move for the turn, the player loses.

Mr. Sheep finds this game might be unfair to him. So Mr. Panda allows Mr. Sheep to choose a connected subgraph of the tree which also contains the vertex 1 along with the token on the vertex. If both Mr. Panda and Mr. Sheep play **optimally**, in how many different ways can Mr. Sheep choose a subgraph for them to play the game on so that he can win? Two ways are considered different if the subgraphs in these two ways differ. As the answer can be very large, you only need to output it modulo $(10^9 + 7)$.

Input

The first line of the input gives the number of test cases, T ($1 \le T \le 10$). T test cases follow.

For each test case, the first line contains an integer n ($1 \le n \le 2 \times 10^5$), representing the number of vertices of the tree.

The next $n-1$ lines each contains two integers x and y ($1 \le x, y \le n$), indicating there is an edge between vertex x and vertex y. It is guaranteed that the given edges form a tree.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from 1) and `y` is the number of different winning ways modulo $(10^9 + 7)$.

Example

standard input	standard output
2 2 1 2 6 1 2 2 3 1 4 4 5 4 6	Case #1: 1 Case #2: 5

Problem H. Mr. Panda and SAD

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

Mr. Panda owns many strings with only uppercase letters. He defines the happiness of a string to be the number of occurrence of "SAD" (SAD stands for Shanghai Algorithm Discussion) in the string.

One day Mr. Panda broke one of his precious string inadvertently into n pieces. The i^{th} piece had the string s_i on it. He collected all the pieces and turned to Mr. Sheep for help. As Mr. Panda was very sad about the broken string, Mr. Sheep comforted him that if they concatenated all the pieces into a new string, the happiness of the string could be even greater than its original happiness value! But they really didn't know how to concatenate pieces to get the maximum happiness. It is your turn to help them again.

Input

The first line of the input gives the number of test cases, T ($1 \le T \le 100$). T test cases follow.

Each test case begins with a line containing one integer n ($1 \le n \le 2 \times 10^5$), the number of pieces Mr. Panda's precious string was broken into.

In the next n lines, each line contains a string s_i ($1 \le |s_i| \le 20$) with only uppercase letters.

It is guaranteed that the sum of n in all cases is not greater than 10^6, and the sum of $|s_i|$ in all cases is not greater than 2×10^6.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from 1) and `y` is the maximum happiness of the string that can be achieved by concatenating the pieces.

Example

standard input	standard output
3 3 SAD D SA 3 SS A DD 4 DS SA ADSA D	Case #1: 2 Case #2: 1 Case #3: 3

Note

For the first sample, the new string can be concatenated as `SADSAD`, hence the happiness is 2.

For the second sample, the new string can be concatenated as `SSADD`, hence the happiness is 1.

For the third sample, the new string can be concatenated as `SA+DS+ADSA+D` = `SADSADSAD`, hence the happiness is 3.

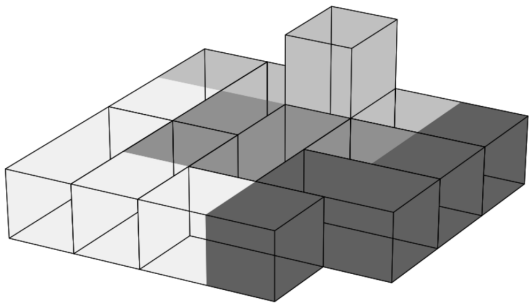
Problem I. Mr. Panda and Blocks

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Mr. Panda recently received a bucket of toy blocks as his birthday gift. Each block is a $1 \times 1 \times 2$ cuboid, which is constructed by a pair of face-to-face $1 \times 1 \times 1$ colored cubes. There are n types of colors, labeled as $1, 2, \ldots, n$.

Mr. Panda checked all of the blocks, and he found that he had just $\frac{n \times (n+1)}{2}$ blocks and each of these blocks was painted with a unique pair of colors. That is, for each pair of colors (i, j) $(1 \le i \le j \le n)$, he had exactly one block with one cube colored i, and the other colored j.

Mr. Panda plans to build a fantastic castle with these blocks today.



Firstly, he defines an attribute called *connected*:

1. If cube A shares a face with cube B, they are connected.
2. If cube A and cube B are connected, and cube B and cube C are connected, then cube A and cube C are connected.
3. If all pairs of cubes in the castle are connected, the castle is connected.

Then he comes up with the following requirements:

1. The whole castle should be connected.
2. For any color i, if only consider the cubes with that color, the sub-castle formed by these cubes should also be connected.

However, after many attempts, Mr. Panda still cannot build such a castle. So he turns to you for help. Could you please help Mr. Panda to build a castle which meets all his requirements?

Input

The first line of the input gives the number of test cases, T $(1 \le T \le 10)$. T test cases follow.

For each test case, one line contains an integer n $(1 \le n \le 200)$, representing the number of colors.

Output

For each test case, first output one line containing "`Case #x:`", where `x` is the test case number (starting from 1).

If it's impossible to build a castle that satisfies Mr. Panda's requirements, output a single line containing "NO" (quotes for clarity).

If it's possible to build the castle, first output a single line containing "YES" (quotes for clarity).

Then, output $\frac{n\times(n+1)}{2}$ lines describing the coordinates of all the blocks. Each of these lines should be outputted in the form of $i, j, x_i, y_i, z_i, x_j, y_j, z_j$ ($1 \le i \le j \le n, 0 \le x_i, y_i, z_i, x_j, y_j, z_j \le 10^9$), which means for the block (i, j), the cube with color i is located at (x_i, y_i, z_i) and the other cube with color j is located at (x_j, y_j, z_j). You should make sure that each pair of (i, j) occurs exactly once in your answer.

In case there is more than one solution, any of them will be accepted.

Example

standard input	standard output
2 3 4	Case #1: YES 1 1 1 1 0 1 2 0 1 2 1 3 0 1 4 0 1 3 2 1 0 3 1 0 2 2 2 2 0 2 3 0 2 3 2 4 0 3 4 0 3 3 3 2 0 3 3 0 Case #2: YES 1 2 1 3 0 1 4 0 1 1 1 2 0 1 1 0 1 3 2 1 0 2 2 0 2 3 2 4 0 2 3 0 1 4 3 1 0 4 1 0 3 3 3 2 0 3 3 0 2 2 3 4 0 3 4 1 4 4 4 2 0 5 2 0 3 4 4 3 0 5 3 0 2 4 4 4 0 5 4 0

Problem K. Russian Dolls on the Christmas Tree

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	1024 mebibytes

Christmas is still one month away, but Mr. Panda already starts the Christmas preparation. Mr. Panda is decorating a Christmas tree with a set of Russian dolls. There are n Russian dolls numbered $1, 2, \ldots, n$. The i^{th} doll is designed to be perfectly nested inside the $(i+1)^{th}$ doll for all $1 \le i \le n-1$. Nesting dolls are stable only if they have neighboring ordinal numbers, otherwise the smaller one will slide out from the larger one. Dolls can be nested recursively. For example, the n dolls can be nested all the way up from smallest to largest until there is only one doll left.

The Christmas tree happens to have n nodes with one Russian doll dangling on each node, where the doll numbered 1 is put at the tree root. Mr. Panda will invite his friend Mr. Sheep to collect some dolls from the Christmas tree as gifts on Christmas Eve. Mr. Sheep will pick a tree node and collect all the dolls in the sub-tree with the selected node as the sub-tree root.

As there could be a lot of dolls, Mr. Sheep want to nest the dolls he collects for easy carrying. He wonders for each tree node, how many dolls will be ended up if he nests them as many as possible. Note that the dolls should be stably nested.

Input

The first line of input gives the number of test cases T ($1 \le T \le 10$). T test cases follow.

Each test case starts with a line containing an integer n ($1 \le n \le 2 \times 10^5$), the number of dolls and also the number of tree nodes.

The next $(n-1)$ lines each contains two integers x and y ($1 \le x, y \le n$), denoting the dolls numbered x and y are neighbors in the Christmas tree.

It is guaranteed that the sum of n in all cases is not greater than 10^6.

Output

For each test case, the output consists one line starts with "`Case #x:`", where `x` is the test case number (starting from 1), followed by next n integers. The i^{th} ($1 \le i \le n$) integer indicates the number of dolls will be ended up if Mr. Sheep selects the tree node that contains the doll numbered i, collects all the dolls in the sub-tree, and nests the dolls stably as many as possible.

Example

standard input	standard output
1 7 1 2 2 4 2 6 1 3 3 5 3 7	Case #1: 1 3 3 1 1 1 1

Problem L. Spiral Matrix

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Lee got a ticket to the Google Developer Day. When he came to the exhibition hall, he found that the booths were located in a matrix of size $n \times m$.

Unfortunately, Lee had badly sprained his left ankle a week before the GDD when playing basketball. As a result, he couldn't wander freely as he wanted. His ankle would get hurt if he tried to turn left. Lee also didn't want to make a big right turn by turning right multiple times, which would make him look stupid.

Lee wanted to visit each booth exactly once. Given his ankle situation, he made some rules visiting the booths. He could start with any booth in the exhibition hall, and choose an initial direction. Then each move would have to follow one of the possible moves:

1. Go straight to visit the next booth.
2. Turn right once and then go straight to visit the next booth.

You are a friend of Lee and you have the good habit of helping him. Can you find out the number of different ways to visit each booth exactly once? Two ways are considered different if and only if the visiting orders in these two ways vary.

Input

The first line of the input gives the number of test cases, T ($1 \le T \le 100$). T test cases follow.

For each test case, the first and only line contains two integers n and m ($1 \le n, m \le 100$), the number of rows and the number of columns of the matrix, respectively.

Output

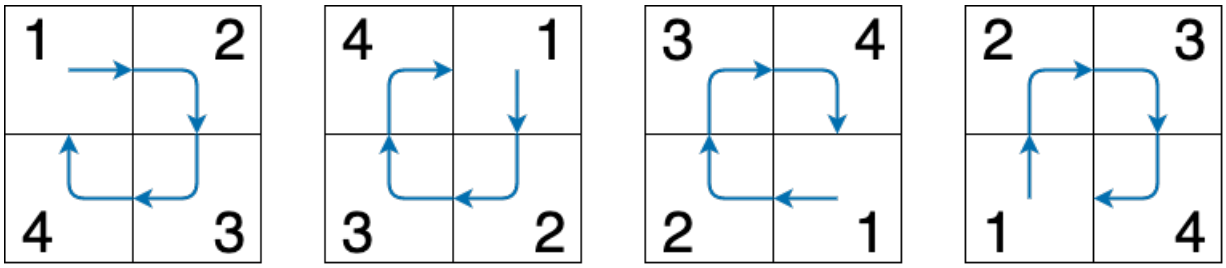
For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from 1), and `y` is the number of different ways modulo $(10^9 + 7)$.

Example

standard input	standard output
1 2 2	Case #1: 4

Note

Here are the 4 different ways for $n = 2, m = 2$.



Problem M. Circle

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

On some display as pixels are used squares with opposite corners (x, y) and $(x+1, y+1)$ for each pair of integers (x, y). A pixel is is lit if it have more than one common point with circle being drawn. Command for drawing a circle have three parameters: two coordinates of center and radius of the circle.

Compute the exact number of pixels that are lit when a circle with a given command for drawing a circle.

Input

Input consists of three integers x, y, and r ($1 \le x, y, r \le 10^6$), specifying respectively the center (x, y) and radius r of the circle.

Output

Print the number of pixels that are lit in the circle, specified by given command.

Example

standard input	standard output
2 2 1	4
6 6 5	88

Problem N. Ecosystem

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 Mebibytes

We will model an ecosystem as follows: there are n species. Each species only eats species lower on the food chain, which means species with lower numbers. For each species, you will be given the initial number of members remaining after some of them have died directly because of the pollution. Furthermore, you will be given their diet: how many members of each of the other species one member needs to eat to survive. That is supposed to describe a "balanced diet": for instance, if we determine that a seal needs to eat a herring and a squid per day, then substituting two herrings and half a squid, or 10 squids, is not sufficient. If the supply of one or more of the food sources is already depleted, then the remaining members of the higher species starve and die (much to the delight of their other prey). For the purpose of this problem, to avoid complication, you will process the species in increasing order. For instance, if both seals (e.g., species 5) and dolphins (e.g., species 12) need herring in their diet, and the seals have eaten all the remaining herrings, then all the dolphins will die.

You are to output the number of surviving members of the species.

Input

The first line contains an integer n ($1 \le n \le 100$), the number of species. This is followed by n lines, with line i describing species i, by giving i integers. The first number is the current population of species i. This is followed by $i - 1$ integers, giving the number of members of species 1, 2, ..., $i - 1$ which a member of species i needs to eat to survive.

Output

Print the number of members of the species (one per line) which survive after one unit of time compared to the starting configuration.

Example

standard input	standard output
6 220 30 3 1020 0 5 310 1 0 1 22 2 0 0 0 11 1 0 0 0 1	69 0 0 6 11 11

Problem O. MMORPG

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

In one MMORPG you can practice with guildmasters and raise your skills: sword mastery for 100 gold and no gems per level, archery for 125 gold and 50 gems per level, two-weapon fighting for 50 gold and 100 gems.

Given your current resources and how much one level in each skill increases your strength in battle, determine the maximum strength increase you can obtain.

Input

Input consists of no more than 60 test cases.

Each test case consists of 5 integers M ($0 \le M \le 5 \times 10^4$), the amount of gold you have, G ($0 \le G \le 5 \times 10^4$), the amount of gas you have, S ($0 \le S \le 1000$), the strength increase for one additional level in sword mastery, A ($0 \le A \le 1000$), the strength increase for one additional level in archery, and T ($0 \le T \le 1000$) — the strength increase for one additional level in two-weapon fighting.

The input is terminated with test case consisting of five zeroes, which shouldn't be processed.

Output

Output a single line containing the strength increase you can obtain.

Example

standard input	standard output
900 800 15 25 35 0 0 0 0 0	355

Problem P. Orienteering Championship

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Bill and Alexey are organizing the orienteering world championship in Moscow in 2020. Since championship is very popular for decades, they has a lot of route proposals. They must find a route with just the right length, but they doesnt have time to measure them all by running through the control points in order, as they are too busy with organization part of the finals. The job is therefore given to Jeff.

As Jeff is slightly less interested in orienteering, he figures out a more time-saving way of measuring the length of all the routes. He brings his GPS, visits all the points of all the routes in the most convenient order, and goes home early to do the rest of the job on his computer.

Input

The first line of input gives $1 \le n \le 1000$, the total number of control points. Then follow n lines giving their coordinates, with two floating-point numbers x_i and y_i, with $0.0 \le x_i, y_i \le 10000.0$. The number of routes is then given by $1 \le m \le 100$. Each route is defined by first a line with $2 \le p \le 17$, the number of control points (including start and goal), and then a line with p indexes $0 \le i < n$, identifying them.

Output

For each test case output the total track distance, rounded, with no decimals.

Example

standard input	standard output
5 0.0 0.0 1000.0 1000.0 123.45 0.0 3475.43 7765.4 4325.9865 13.0 2 2 0 1 4 3 1 4 0	1414 14999

Problem Q. Parallel Computing

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

The Parallel Software Foundation did the next experiment: for a number of problems where a serial implementation is available, it estimates the number of times this program will be executed within the next year (when the program will probably be rewritten).

Then members of Foundation develop a parallel version of the program and measure the amount of time they used developing it. At the end, they measure the execution time of both the parallel and the serial version. Based on the collected data, Foundation wants to know in which cases the overall number of man-hours spent on this program is minimized by parallelizing. Man-hours are spent developing the parallel version and waiting while programs execute.

Input

The input has $t \le 1000$ cases, where t is given by the first line of input. Each test case is given by a line with four integers d, n, s and p separated by a single space. d is the time spent developing the parallel version ($0 \le d \le 10^6$), and n is the expected number of times this program will be executed during the next year ($0 \le n \le 10^5$). s and p are the running times of the serial and parallel version of the program respectively ($0 \le s, p \le 1000$).

Output

For each test case output "`parallelize`" on a single line if it is beneficial to parallelize. If it is not beneficial to parallelize, the outputted line should be "`do not parallelize`". If the expected total time spent with the program is similar regardless of whether it is parallelized or not, the outputted line should be "`does not matter`".

Example

standard input	standard output
3 10 2 3 2 20 5 8 2 0 2 1 1	do not parallelize parallelize does not matter

Problem R. Royal Team

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

The famous Royal Soccer team wants to biy new player. Usually this means that they need most expensive player, So they are calling all soccer clubs and asking for their most expensive player. Your task is to help Royal Team find the most expensive player from a list.

Input

The input has $n \leq 100$ cases, where n is given by the first line of input. The first line of each test case is a single positive integer, $p \leq 100$, giving the number of players to consider. Then follow p lines, where each line represents a player. The line starts with a positive integer $c_i < 2 \cdot 10^9$, the price of player i. Then follows a single space before the name of the player. All player prices are unique. Player names are never more than 20 characters long, and contain no spaces.

Output

For each test case your program should output a single line giving the name of the most expensive player.

Example

standard input	standard output
2 3 999999 Kane 1000000 Ronaldo 2000000 Messi 2 1000000 Maradona 999999 Pele	Messi Maradona