Problem A. Alien invaders

Input file:	*standard input*
Output file:	*standard output*
Time limit:	10 seconds
Memory limit:	512 mebibytes

The aliens from outer space have (finally!) invaded Earth. Defend yourself, or be disintegrated! Or assimilated. Or eaten. We are not yet sure.

The aliens follow a known attack pattern. There are n attackers, the i-th one appears at time a_i, at distance d_i from you. He must be destroyed no later than at time b_i, or else he will fire his weapon, which will definitely end the fight.

Your weapon is an area-blaster, which can be set to any given power. If fired with power R, it momentarily destroys all aliens at distance R or smaller. It also consumes R fuel cells.

Determine the minimal cost (measured in fuel cells) of destroying all the aliens, without being killed.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

Each test case starts with a line containing the number of aliens n ($1 \le n \le 300$). Of the next n lines, the i-th one contains three integers a_i, b_i, d_i, ($1 \le a_i < b_i \le 10\,000$; $1 \le d_i \le 10\,000$). The i-th alien appears at time a_i, is idle until b_i, and his distance from you is d_i.

Output

For each test case, output one line containing the minimum number of cells needed to destroy all the aliens.

Example

standard input	standard output
1 3 1 4 4 4 7 5 3 4 7	7

Problem B. Bioengineering

Input file:	*standard input*
Output file:	*standard output*
Time limit:	29.4 seconds
Memory limit:	512 mebibytes

Viruses are usually bad for your health. How about fighting them with... other viruses? In this problem, you need to find out how to synthesize such good viruses.

We have prepared for you a set of strings of the letters '`A`', '`G`', '`T`' and '`C`'. They correspond to the DNA nucleotide sequences of viruses that we want to synthesize, using the following operations:

- Adding a nucleotide either to the beginning or the end of the existing sequence.
- Replicating the sequence, reversing the copied piece and gluing it with the original (so that e.g., "`AGTC`" becomes "`AGTCCTGA`").

We're concerned about efficiency, since we have very many such sequences, some of them very long. Find a way to synthesize them in a minimum number of operations.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

Each test case consists of a single line containing a non-empty string. The string uses only the capital letters '`A`', '`C`', '`G`' and '`T`' and is not longer than 100 000 characters.

Output

For each test case, output a single line containing the minimum total number of operations necessary to construct the given sequence.

Example

standard input	standard output
4 AAAA AGCTTGCA AAGGGGAAGGGGAA AAACAGTCCTGACAAAAAAAAAAAAC	3 8 6 18

Problem C. City of Eternal Festivities

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2.4 seconds
Memory limit:	512 mebibytes

In The City of Eternal Festivities, there are n street junctions and $n-1$ bidirectional streets, each street connecting two of the junctions. Between every two junctions, there is exactly one (direct or indirect) path connecting them. No junction is an endpoint for more than 10 streets.

Every 13th of September (the 256th day of the year), there are many festivities going on in The City. In particular, the citizens want to organize m parades. The parade number i starts at junction u_i and ends at v_i, following the unique path between the endpoints.

As the mayor of The City, you are responsible for citizens' safety. Therefore you decreed that no two parades are ever allowed to use the same street, though they can have common junctions, or even common endpoints.

To appease your citizens, try to organize as many parades as possible, without breaking the safety regulations.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

The first line of each test case contains a single integer: the number of junctions n ($2 \le n \le 1000$). Each of the next $n-1$ lines contains two integers a, b ($1 \le a \ne b \le n$), denoting that junctions a and b are connected by a street. Each junction has at most 10 streets leaving it.

The next line contains a single integer: the number of planned parades m, $0 \le m \le \binom{n}{2}$. Each of the next m lines contains two integers u_i, v_i ($1 \le u_i \ne v_i \le n$), meaning that a parade is planned to start at junction u_i, and finish at junction v_i. No two parades share both endpoints.

Output

For each test case, output one line containing the largest number of parades that can be organized with no street used by more than one parade.

Example

standard input	standard output
1 6 1 2 2 3 3 4 3 5 3 6 4 1 3 4 5 5 6 6 4	2

Problem D. Dictionary

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3.1 seconds
Memory limit:	512 mebibytes

According to a popular belief, computer programmers drink a lot of coffee and know only a few words. The vocabulary of a typical programmer consists of just three words. Besides, he rarely knows how to spell them. To help programmers with their spelling mistakes, we published a book titled «The Dictionary of the Three Words Every Typical Programmer Should Know».

You got a copy of the book but, soon after that, you spilled your coffee over it. Now, you cannot read some of the characters. Fortunately, the three words were, as usually in dictionaries, distinct and printed in lexicographical order.

Before you attempt to use that fact to recover the missing characters, you want to know in how many different ways you can do it. Since you expect this number might be large, you want to know it modulo $10^9 + 9$.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

Each test case consists of three lines, each containing a single nonempty word – in the order they appear in the dictionary. Words consist of small letters of the English alphabet and quotation marks, the latter denoting missing characters. Each word is at most 1 000 000 characters long.

Output

For each test case, output one line containing the number of different ways you can substitute each question mark with one of the 26 letters from '`a`' to '`z`' in such a way that the three words are distinct and in lexicographical order. The number should be printed modulo $10^9 + 9$.

Example

standard input	standard output
3 ?heoret?cal c?mputer ?cience jagiellonian ?niversity kra?ow ? b c	42562 52 1

Problem E. Express As The Sum

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

Given an integer N, express it as the sum of at least two consecutive positive integers. For example:

$10 = 1 + 2 + 3 + 4$

$24 = 7 + 8 + 9$

If there are multiple solutions, output the one with the smallest possible number of summands.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

Each test case consists of one line containing an integer N ($1 \le N \le 10^9$).

Output

For each test case, output a single line containing the equation in the format:

`N = a + (a+1) + ...+ b`

as in the example. If there is no solution, output a single word "`IMPOSSIBLE`" instead.

Example

standard input	standard output
3 8 10 24	IMPOSSIBLE 10 = 1 + 2 + 3 + 4 24 = 7 + 8 + 9

Problem F. Factory

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

A very important and complicated machine on the factory consists of n wheels, numbered $1, 2, \ldots, n$. They are actually cogwheels, but the cogs are so small that we can model them as circles on the plane. Every wheel can spin around its center.

Two wheels cannot overlap (they do not have common interior points), but they can touch. If two wheels touch each other and one of them rotates, the other one spins as well, as their micro-cogs are locked together.

A force is put to wheel 1 (and to no other wheel), making it rotate at the rate of exactly one turn per minute, clockwise. Compute the rates of other wheels' movement. You may assume that the machine is not jammed (the movement is physically possible).

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

Each test case consists of one line containing the number of wheels n ($1 \le n \le 1000$) . Each of the following lines contain three integers x, y and r ($-10\,000 \le x, y \le 10\,000$; $1 \le r \le 10\,000$) where (x, y) denote the Cartesian coordinates of the wheel's center and r is its radius.

Output

For each test case, output n lines, each describing the movement of one wheel, in the same order as in the input. For every wheel, output either "`p/q clockwise`" or "`p/q counterclockwise`", where the irreducible fraction p/q is the number of wheel turns per minute. If q is 1, output just p as an integer. If a wheel is standing still, output "`not moving`".

Example

standard input	standard output
1 5 0 0 6 6 8 4 -9 0 3 6 16 4 0 -11 4	1 clockwise 3/2 counterclockwise 2 counterclockwise 3/2 clockwise not moving

Problem G. Game 2048

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6.5 seconds
Memory limit:	512 mebibytes

Some computer games are extremely fun and this problem may be about one of these.

You are given a sequence of one dimensional blocks, each of length that is a power of two. The goal of the game is to merge all the blocks into one big block. The blocks are presented one by one, and for each one you have to decide whether to stick it immediately to the left or immediately to the right of the previous blocks.

Every time two blocks of the same size are adjacent, they merge into one block that is twice as long as each of them. Note that, as long as possible, the resulting blocks immediately merge with adjacent ones. For example, if the current sequence of blocks is 2, 4, 16, then sticking 2 on the left side leads to 8, 16, while sticking it on the right side gives 2, 4, 16, 2. Note that at any moment there is at most one mergeable pair of blocks.

You have lost the game (again!) and you are wondering whether there was any way to win at all. Analyze the sequence to find out.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

Each test case consists of two lines. The first one contains a positive integer $n \leq 1\,000$ – the length of the sequence. The next line contains a sequence of n block lengths, each a power of two. The sum of all the lengths does not exceed 2^{13}.

Output

For each test case, output one line containing the word "`no`" if winning the game is not possible. Otherwise, output a word consisting of n letters '`l`' and '`r`', denoting whether the corresponding block should be stuck to the left or to the right. Note that for the first block it does not matter.

Example

standard input	standard output
3 9 2 8 4 1 1 4 4 4 4 5 2 16 4 8 2 3 2 2 2	rrrlllrrr no no

Problem H. How to Deal With Imp

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

You arrive in Ye Olde Magic Shoppe with some hard-earned gold to purchase wondrous and unique magic items. There are n such items in the shop, each of them locked in a special magic box. The i-th box costs c_i gold pieces to buy, and contains an item worth v_i gold pieces. The costs and item values are known to you, as you have previously read, mastered, and memorized Ye Olde Magic Catalogue.

A mortal, such as you, can safely carry only one magic item. You therefore aim to get the most precious one. And obtain it you would, if not for a malicious, magical creature, known as The Imp.

The Imp can cast a mischievous spell, which transforms the content of any magic box into worthless dust. Of course, he will use the spell just after you buy a box, to make you pay for the item and not get it. You are thus forced to buy another box, and then the next one...

The Imp has enough magic to cast the spell at most k times. He can, of course, refrain from using it, allowing you to keep an item. You can walk away at any time, empty-handed (though it would surely be a disgrace). However, if you get an item, you must keep it and leave the shop. You aim to maximize your gain (the value of the acquired item minus all the expenses paid previously), while The Imp wants to minimize it. If both you and the creature use the optimal strategy, how much gold will you earn?

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

Each test case starts with a line containing the number of items n ($1 \le n \le 150\,000$) and the the maximum number of The Imp's spells k ($0 \le k \le 9$). The next n lines contain the items' values and costs, the i-th line containing the numbers v_i and c_i, in that order ($0 \le v_i, c_i \le 10^6$).

Output

For each test case, output one line containing your gain.

Example

standard input	standard output
1 3 1 10 5 8 1 20 12	7

Problem I. Infrastructure and Committees

Input file:	*standard input*
Output file:	*standard output*
Time limit:	20.6 seconds
Memory limit:	512 mebibytes

Winning the election was simpler than you expected: it was enough to promise to finally build a good quality, country-wide road infrastructure, of course without crippling the budget... Your happiness did not last long, however: it seems, that the citizens have found a way to actually hold you accountable for your promise!

There are n major cities in your country. The Ministry of Transport has prepared a detailed map, outlining m possible highway connections, together with their costs. The Quality Assurance Committee will not let you build a highway cheaper than l, and the National Spendings Regulatory Committee will not let you build a highway more expensive than h. To claim a "country-wide" network, you have to connect (possibly indirectly) as many pairs of cities, as it is possible within these two constraints. You have to find the cheapest way to do it, and you have to find it quickly! Of all networks that meet the constraints and connect the most pairs of cities, compute the cost of the cheapest one.

To make things worse, both committees are heavily influenced by your angry competitors: each time you publish your hard-prepared plan, they immediately change their rulings l and h, and you are forced to start from scratch.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

The first line of each test case contains integers n and m ($1 \le n \le 1000$; $0 \le m \le 100\,000$) – the number of cities in the country, and of possible direct connections, respectively.

Each of the following m lines contains three integers x, y, w ($1 \le x \ne y \le n$; $1 \le w \le 1\,000\,000$), denoting that the cities x and y can be connected by a bidirectional highway at cost w. There might be many ways to connect a single pair of cities.

The following line contains an integer q ($1 \le q \le 1\,000\,000$) – the number of rulings of the committees. Each of the following q lines contains two integers. The first of the lines contains the initial rulings l_1, h_1, given directly. The rest of the rulings are encoded. The numbers in the j-th of the lines for $j > 1$ are $l_j + c_{j-1}$ and $h_j + c_{j-1}$, where l_j and h_j are the actual rulings and c_{j-1} is the correct answer for the preceding rulings l_{j-1} and h_{j-1}.

All rulings satisfy $1 \le l_j \le h_j \le 1\,000\,000$.

Output

For each test case, output q lines, one for each ruling. In the j-th of them, output the minimal cost c_j of building a highway network which adheres to the committees' constraints, and creates the maximum number of connected pairs of cities.

Example

standard input	standard output
1 5 7 1 2 2 2 3 4 3 4 3 4 5 1 5 1 3 2 5 4 1 4 5 5 1 2 4 7 11 12 11 13 18 19	3 9 8 14 13

Note

The actual rulings of the committees are

$(1, 2)$, $(1, 4)$, $(2, 3)$, $(3, 5)$ and $(4, 5)$.

The cheapest highway networks adhering to these rulings consist of connections

$\{(1, 2), (4, 5)\}$, $\{(2, 1), (1, 5), (5, 4), (4, 3)\}$, $\{(1, 2), (1, 5), (3, 4)\}$, $\{(1, 5), (5, 2), (2, 3), (3, 4)\}$ and $\{(3, 2), (2, 5), (1, 4)\}$, respectively.

Problem J. Joining The Bricks Together

Input file:	*standard input*
Output file:	*standard output*
Time limit:	10.7 seconds
Memory limit:	512 mebibytes

You are given a sequence of white (W) and black (B) bricks. The goal is to partition it into some number of non-empty, contiguous blocks, each one having the same ratio of white and black bricks.

Of course one can always «partition» the sequence into one single block (which is not very interesting). We want, however, to have as many blocks as possible. Consider for example the following sequences and its partitions:

`BWWWBB = BW + WWBB` (ratio 1:1),

`WWWBBBWWWWWWWWWB = WWWB + BBWWWWWW + WWWB` (ratio 3:1).

Note that both of these partitions are optimal with respect to the number of blocks.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

Each test case starts with one line containing an integer n ($1 \le n \le 10^5$) which is the length of the description of a sequence. Each of the following n lines consists of an integer k ($1 \le k \le 10^9$) and one of the characters `W` or `B`, meaning that k bricks of the given color follow next in the sequence. It is guaranteed that the total length of the brick sequence does not exceed 10^9.

Output

For each test case, output a single line containing the largest possible number of blocks.

Example

standard input	standard output
3 3 1 B 3 W 2 B 4 3 W 3 B 9 W 1 B 2 2 W 3 W	2 3 5

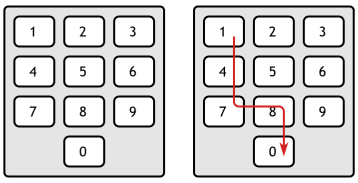
Problem K. Keyboard Troubles

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

Good morning! This is your 5am wake-up call! A partly cloudy day is expected with light rain coming afternoon...

You have just woken up. You desperately need coffee... and... more coffee... and some cereal. And your clothes. And coffee.

To prepare warm cereal, you put some milk into a microwave, trying to heat it for k seconds. You must enter k on the microwave keyboard:



As you still haven't had your coffee, your hand (along with eyes and brain) keeps falling down. You are only able to enter a number if your hand would only move downwards and/or to the right. You cannot go back left, nor move your hand up, though you can press the same key again. And again... and again...

For example, you can enter the number 180 or 49, but not 98 or 132. Enter a number that is as close to k as possible. If there are two solutions, enter any one of them. You are too sleepy to actually care. And you need coffee.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

Each test case consists of one line containing an integer k ($1 \le k \le 200$).

Output

For each test case, output a number that is closest to k which can be entered on the keyboard.

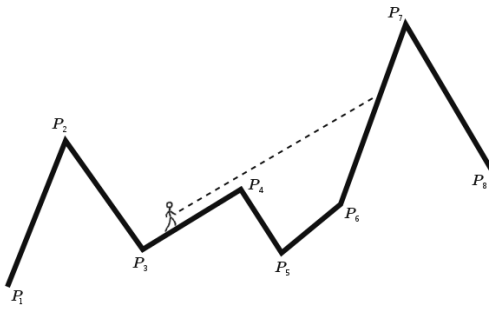
Examples

standard input	standard output
3 180 83 132	180 80 133

Problem L. Landscape

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6.9 seconds
Memory limit:	512 mebibytes

You travel through a scenic landscape consisting mostly of mountains – there are n landmarks (peaks and valleys) on your path. You pause for breath and wonder: which mountain are you currently seeing on the horizon?



Formally: you are given a polygonal chain $P_1P_2 \dots P_n$ in the plane. The x coordinates of the points are in strictly increasing order. For each segment P_iP_{i+1} of this chain, find the smallest index $j > i$, for which at least one point of P_jP_{j+1} is visible from P_iP_{i+1} (lies **strictly above** the ray $P_iP_{i+1}^{\rightarrow}$).

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

The first line of each test case contains an integer n ($2 \le n \le 100\,000$) – the number of vertices on the chain.

Each of the following n lines contains integer coordinates x_i, y_i of the vertex P_i ($0 \le x_1 < x_2 < \ldots < x_n \le 10^9$; $0 \le y_i \le 10^9$).

Output

For each test case, output a single line containing $n - 1$ space-separated integers. These should be the smallest indices of chain segments visible to the right, or 0 when no such segment exists.

Example

standard input	standard output
2 8 0 0 3 7 6 2 9 4 11 2 13 3 17 13 20 7 7 0 2 1 2 3 1 4 0 5 2 6 1 7 3	0 3 6 5 6 0 0 6 4 4 0 6 0