Algospot.com 2011 ACM-ICPC Daejeon Regional Warm-up Contest

October 29, 2011

Problems A to J, Total 18 pages

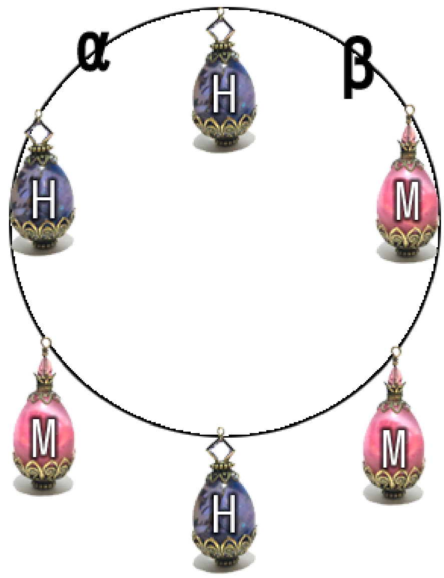
This page is intentionally left blank.

Problem A. Final Exam

There's a bead art course in University of Algospot. The course is notorious for its tough exam, especially the final one, in which you should handle threaded, colored beads. Every bead is colored in one of 26 colors, denoted by capital alphabets 'A' to 'Z'. So we can represent a thread of colored beads by a concatenation of their colors. Such expression is called an **AlgoBead representation**.

You can make a **necklace** by connecting the two ends of a thread. You can find an **AlgoBead representation** of a necklace by cutting the necklace somewhere and reading the colors of the resulting thread. Therefore, a single necklace can have different **AlgoBead representation**s depending on where to cut the necklace and which direction to read the colors.

For example, see the following necklace.



You can cut the necklace at α to get either 'H-M-M-H-M-H' or 'H-M-H-M-M-H' as the **AlgoBead representation** of the necklace, depending on the direction you read the colors. If you cut at β and read counterclockwise, the representation would be 'H-H-M-H-M-M', which is the lexicographically smallest **AlgoBead representation** among all possibilities.

In the actual final exam, you have to solve the following problem. You're given a long thread of beads. You will cut out K consecutive beads from the thread and connect both ends to make it a necklace. Your goal is to find the lexicographically smallest **AlgoBead representation** among all possible necklaces.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

Each test case is given as a single line, and it contains an integer K and a string which is the **AlgoBead representation** of the given thread of beads. The given string consists of only uppercase alphabet letters from 'A' to 'Z'. The length of the given string is at least 1 and at most 5000. K is at least 1 and is not greater than the length of the given string.

Output

For each test case, print exactly one line containing a string which should be the lexicographically smallest **AlgoBead representation** as explained in the problem statement.

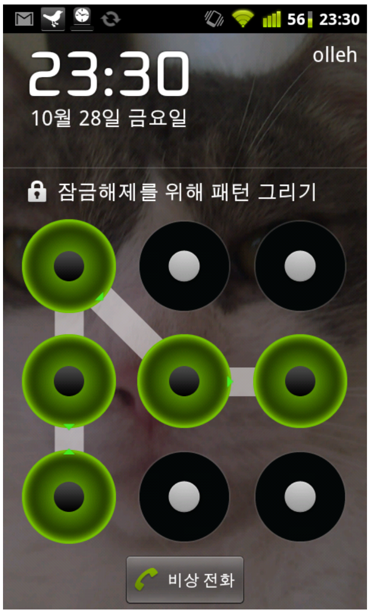
Sample input and output

Standard Input	Standard Output
2 6 DACCACACA 1 ALGOSPOT	AACACC A

Problem B. Pattern Lock

One of the famous feature of Android™ is a pattern−based lock screen. There are nine dots displayed in the screen, aligned in a three by three grid in equal distances. A password is a visual pattern consisting of several segments connecting the dots.

For simplicity, we number the dots in the topmost row as 1, 2, and 3 from the left to the right, 4 to 6 for the second row, ditto. To input a pattern, an user starts by touching one of the dots by her finger. The selected dot is then colored as green, and the other dots remain as gray. From there she can drag around the screen without detaching the finger, and when her finger touches another gray dot it is added to the pattern: the color changes into green, and a line segment connecting between the lastly added green dot and the current one is added. The pattern is finalized when she detach her finger from the screen. Moreover, it is not allowed to draw a segment containing any gray dot. For example, it is prohibited to use a password $\langle 1,3,2 \rangle$ since it is not allowed to draw a segment from 1 to 3 which contains an unselected dot 2. The following figure is an example of a good password.



So the password above corresponds to the sequence $\langle 4,7,1,5,6 \rangle$ — please note that there are some small arrows representing the direction of the next dot (it doesn't matter in this problem even if you cannot see the arrow).

xhae wants to crack the password for ainu7's android phone, so he watched the pattern behind the shoulder — he figured out the shape of the password but he doesn't remember the exact sequence of the password. To be exact, those small arrows are too small to recognize from behind, so he just remember the overall shape of segments connecting green dots.

Now he is curious about the number of different sequences which leads to the same shape. For instance, both $\langle 7,4,1,5,6 \rangle$ and $\langle 4,7,1,5,6 \rangle$ leads to the same shape as the figure above. So there are at least two different sequences with the same shape in this case. Write a program to help him.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case will contain the number of selected dots, N ($N > 1$). The following line will contain space-separated N integers denoting a valid password sequence for ainu7's android phone.

Output

Print exactly one line for each test case. The line should contain the number of sequences which leads to the same shape as input sequence.

Sample input and output

Standard Input	Standard Output
2 4 1 2 6 9 5 1 2 5 8 9	2 2

Problem C. poPOWERwer

Recently, Koreans started using an Internet slang which is in the form of "po−WORD−wer" to emphasize the meaning or impact of the word placed in the middle.

For instance, when one reads a blog post about a very difficult Dynamic Programming problem on Algospot.com, he could simply post a comment that reads "poDPwer." If one is going to be on a long vacation, she may write "poVACATIONwer" on her facebook wall. As you can see, any word could be written in this power−form.

One day poLEWHA0wer became curious about the origination of this slang. But his interest somehow ended up with a rather different, mathematical poPOWERwer, x^y.

Specifically, he wants to find out the number of different powers that are of the form x^y, where $1 \leq x \leq a$ and $1 \leq y \leq b$. For example, there are six different numbers of the form x^y if $a = 4$ and $b = 2$: $\{1, 2, 3, 4, 9, 16\}$.

Write a poPROGRAMwer that calculates the number of different powers x^y.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The one and only line of each test case will contain two integers a $(1 \leq a \leq 10)$ and b $(1 \leq b \leq 100)$.

Output

Print exactly one line for each test case. The line should contain the number of different powers of the form x^y.

Sample input and output

Standard Input	Standard Output
3 2 3 4 2 7 5	4 6 29

Problem D. Nim_N Game

Nim is a well known two−player mathematical game. The rule of the game is simple. The game begins with a certain number of stones divided into H separated heaps. Two players take turns alternatively. In each turn, a player must choose a nonempty heap and remove some stones from it. She can remove any number of stones, as long as the selected heap has enough stones. A player loses when she cannot remove any stones (i.e. there are no stones left). We will denote the state of the game as an integer sequence $\langle a_1, a_2, \cdots, a_H \rangle$ − it represents H heaps, ith of which containing a_i stones.

It is known that the result of the game is determined if each player always plays "optimally". For example, in the game $\langle 2, 1 \rangle$, the first player will always win if played optimally: there is a strategy for the first player to win the game no matter what the second player does. The strategy goes like this: the first player removes one stone from the first heap, changing the game as $\langle 1, 1 \rangle$. Regardless of the second player's move, the first player takes the last stone.

If a move guarantees a player's victory (even if the other player does her best), we call it a "winning move". For the game described above, taking one stone from the first heap is a winning move.

Since it is a famous game, there are lots of variants of Nim. Nim_N is one of such games. The rule of Nim_N is as follows:

- Nim_0 is the same game as "normal" Nim game described above.
- Nim_N where the binary expansion $N = b_0 2^0 + b_1 2^1 + \cdots$ $(0 \le bi \le 1$, b_i integer) is almost the same game as Nim_0 with an additional rule: "It is not allowed to make a winning move in Nim_i if b_i is nonzero". For example,
  - Nim_1 disallows a move that is a winning move in Nim_0.
  - Nim_2 disallows a move that is a winning move in Nim_1.
  - Nim_3 disallows a move that is a winning move in Nim_0 or Nim_1.

Given a Nim_N game, write a program that computes the winner of the game if both player plays optimally.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case will contain two integers N $(0 \le N \le 2^{31} - 1)$ and the number of heaps H $(1 \le H \le 100)$.

The second line of each test case will contain H integers denoting the number of stones in each heap a_i $(1 \le a_i \le 2^{31} - 1)$.

Output

Print exactly one line for each test case. The line should contain "`First`" if the first player will win. Print "`Second`" otherwise.

Sample input and output

Standard Input	Standard Output
3 0 2 2 1 1 2 2 1 1 4 3 2 1 1	First First Second

Problem E. Prof. Chwa and his magicCircle class

Go to Wolpyeong station. Stand on the Gate 9. Move 76.8cm to left. Close your eyes, and now dive to the huge and beautiful column in front of you. Then bang, you're now on the train heading for KAIST, the number one magic academy in Korea.

Among the magic classes, the magicCircle class of Prof. Chwa is always one of the most popular classes. Drawing a magicCircle is fun, and joining mCPC is exciting. Many famous magicians were the first on this class such as `astein`, `ryuwonha`, `RR`s, and more. Some of them are under age 25 until now, but I trust they would be great magicians who' ll lead the Korea magic society, once they reach age 25.

But everyone can't be happy. If some loves, then some hates. While students really love the class, Mr. Josh, TA of the magicCircle class, is tired of his work, especially when he returned to his lab and found the message:

"*New homework is on your e-owl. Make a reference solution. Chwa*"

Now it's our turn to help him, since we love you Josh. :D

The new homework is simple. We have some magicPoints on the magicGrid, and some magicPoints are connected, which means that two different magicPoints are linked via a single magicLine. And here's the definition of magicCircle: if we can travel magicGrid starting from an arbitrarily chosen magicPoint, moving via magicLines directly connected to the current magicPoint, and returning to the start magicPoint without using same magicLines nor visiting magicPoints twice except the first one, we call that travel line a magicCircle.

There's no magicCircle from the given magicPoints and magicLines at the moment, and the goal is to make a magicCircle with **length** K by adding minimum number of magicLines on the magicGrid. **Length** of a magicCircle is defined to be the number of magicLines on the magicCircle. Calculate the minimum number of additional magicLines to reach the solution.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case will contain three integers the number of magicPoints N ($3 \leq N \leq 100$), the number of magicLines M ($0 \leq M < N$), and K ($3 \leq K \leq N$). The next M lines will contain two integers p and q ($1 \leq p, q \leq N$), denoting the unique identifier of magicPoints which are connected directly via a magicLine.

Output

Print exactly one line for each test case. The line should contain the minimum number of additional magicLines to make a magicCircle with **length** K.

Sample input and output

Standard Input	Standard Output
1 5 3 4 1 2 2 3 4 5	2

Problem F. Unordered Subsequence 2

The sequence is called unordered if it is neither non−decreasing nor non−increasing. A sequence $\langle a_0, a_1, \cdots, a_{n-1} \rangle$ is called non−decreasing if $a_i \leq a_{i+1}$ for all $i \in [0, n-2]$. Non−increasing sequence is defined in a similar manner. For example, sequence $\langle 1, 2, 7, 7, 10 \rangle$ is non−decreasing and $\langle 3, 2, 2 \rangle$ is non−increasing. The sequence $\langle 2, 2 \rangle$ is both non−decreasing and non−increasing sequence; but the sequence $\langle 1, 3, 3, 1 \rangle$ is neither of them, i.e. unordered. Given a sequence of numbers, you are to find the number of subsequence which is unordered.

A subsequence is a sequence that can be made from the given sequence by deleting zero or more elements without changing the order of the remaining elements.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

Each test case consists of two lines. The first line of each test case contains one integer N $(1 \leq N \leq 2,000)$. The second line contains N space−separated integers — the given sequence. All numbers in this sequence do not exceed 10^6 in absoulte value.

Output

Print exactly one line for each test case. The line should contain the number of unordered subsequence modulo $1,000,000,009$.

Sample input and output

Standard Input	Standard Output
3 5 67 499 600 42 23 3 1 2 3 3 2 3 1	12 0 1

Problem G. Cubes Incognitanus, the liar

"Madoka, Madoka! Be the magical girl, save the world!"

*— from "Gorgeous and fantastic words of 108 greatest liars
who amazed the world 2nd edition,
published with great aids of our
generous and humorous major of Huyuki,
the city where the greatest war occurs."*

Caution: All the descriptions on this problem are fiction, having absolutely no relationship with the real person or places. Also please make sure your room is bright enough, and read the paper with the certain distance apart from it.

Hi father. I'm Jenny Titor, a time traveller from 2036, and your daughter at the same time. Just never mind about my family name, there's a sad legend. You don't have to believe it, but just stop trolling and listen to me for a while.

Did I tell you we're living in the parallel world? Yes, we are. The famous theory. Lots of worlds are now progressing on parallel way, and even now the number of world is increasing more and more with the each choices on every time you make. And each world have it's own value called **divergence**, a non-negative integer.

And now, father, this is a tough part. With the great help from Mid-Childa, the famous archaeologist agency, we finally excavated an ancient book with worlds' **divergence** relation map. This map contains M information that $D_{p_i} \geq D_{q_i} + K_i$ ($0 \leq p_i, q_i < N$, $0 \leq i < M$, $0 \leq K_i \leq 10,000$, p_i, q_i, i, and K_i are integer, D_{p_i} and D_{q_i} stands for the **divergence** of p_i-th and q_i-th world each), meaning p_i-th world's **divergence** is equal or greater compare to the addition of **divergence** from q_i-th world and K_i, the constant.

Yes, it's a good thing. The map toward the Pandora, I love it. The problem is on it's last page. It said "Writer: Cubes Incognitanus". Oh, how can I forget that mean name, the greatest liar ever on every time and world, slaying thousands of magical girls. Rest in peace, Madoka, Mami, Steve, Dennis, John...

So I'm here to ask your help, you SUPER HACKER. Please check whether this map has logical error or not, and if not, tell me a possible **divergence** status for each world. If there's many, tell me the first one on dictionary arrangement so that I can understand it easily. Love you dad, S2. El psy congroo.

Input

The input consists of T test cases. The first line of the input contains an integer T.

THe first line of each test case will contain two integers N and M, where $N \leq 50,000$ and $M \leq 200,000$. Then M lines follow, each containing three integers p_i, q_i and K_i ($0 \leq i < M$).

Output

Print exactly one line for each test case.

If no logical error found on this map, output a possible **divergence** status for each world in a single line;

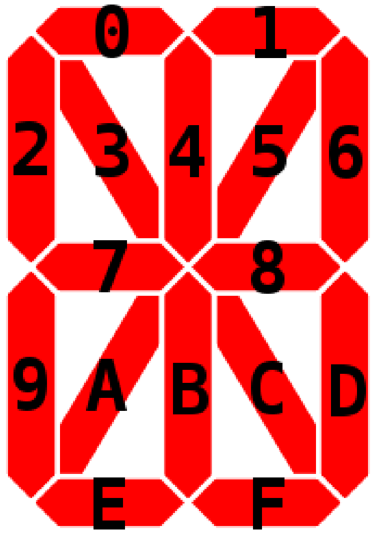
the line should contain N integers $D_0, D_1, \cdots, D_{n-1}$ seperated by a single whitespace. If there are more than one possible **divergence** statuses, you should find the lexicographically smallest one.

If there is any logical error (that is, no possible **divergence** status), print '-1' (without quotes) in the line instead.

Sample input and output

Standard Input	Standard Output
4 2 1 0 1 3 4 2 0 1 600 1 2 600 3 3 0 1 1 0 2 2 1 2 3 3 3 0 1 1 1 2 1 2 0 1	3 0 1200 600 0 0 4 3 0 -1

Problem H. Strobogrammatic Numbers

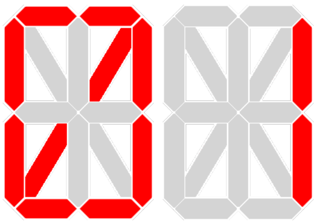


Clever cats Ruby and Jerry are learning some hardware programmings from their human housemate Wonha. They are combining several digital logic gates to show some numbers on sixteen−segment displays.

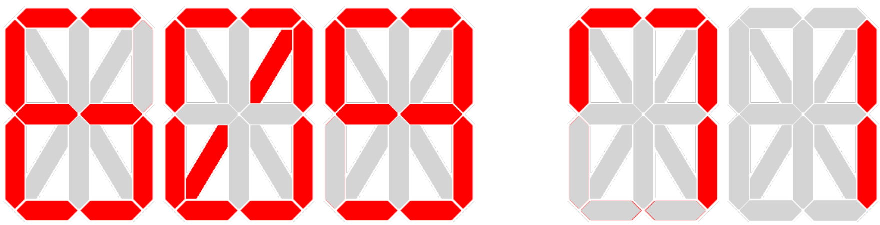
Sixteen−segment display is a display device with 16 line segments that can be individually turned on and off to show alphanumeric characters. You can see the position and identifier of each segment in the figure to the right.

Of course, cats do not necessarily use Arabic numeral system or base 10 as we do. In this problem, cats use a base B number system, each of those B digits is represented by different segment configurations.

A segment configuration tells us which segment to turn on and which to turn off. It is given as a 16−bit number. If the highest bit of the number is on (1), it means segment F will be turned on. The next bit tells us about segment E, and so on. For example, two configurations $1110011001100111_{(2)} = 58983$ and $0010000001000000_{(2)} = 8256$ are conventionally used by humanity to denote 0 and 1, and are shown below.



Ruby and Jerry implemented the actual display without a problem (they are clever). After that, they've got interested in numbers that look same when rotated 180 degrees. For example, the first number in the following example looks the same if put upside down, but the second number doesn't.



Please note that we do not allow horizontal or vertical shift: therefore, configuration given as 8256 (which denotes Arabic numeral 1) is **not** identical to itself when rotated.

These kind of numbers are formally called **strobogrammatic** numbers. Write a program that given segment configurations of B digits, find the K−th smallest strobogrammatic number in this system.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case will contain two integers B $(2 \le B \le 10)$ and K $(1 \le K \le 100)$. B integers separated by spaces representing configurations for each digit 0 from $B-1$ will be given in the next line.

- All configuration will have at least one segment turned on.
- No two segment configurations in a single test case will be equal.

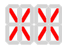
Output

Print exactly one line for each test case, containing K−th smallest strobogrammatic number written in base B. If there is no such number, print "−1" instead.

Sample input and output

Standard Input	Standard Output
2 2 2 43090 2064 3 6 43090 2064 5160	11 121

Notes

For the second test case, these are six smallest strobogrammatic numbers: ,  ,  ,  ,  , and  .

Problem I. Poktanju

"Poktanju" is one of the most distinguishing drinking custom among the Koreans. "Poktan" in Korean means bomb, and "Ju" means alcoholic beverage. Combined together, it means Boilermaker (a kind of beer cocktail), but many Koreans use this expression Poktanju to refer to any mixture of alcoholic beverages. Some well-known Poktanju recipes include So-maek, Gojingamraeju, and Sobaeksanmaek. The picture on the right is an example of how Poktanju is made — shot glasses on the top would be pushed into the beer glasses at the bottom to make up glasses of Poktanju.

Hooyeon (a.k.a. ltdtl) recently came to Korea, and now he is hosting a welcome drinking party. The party consists of N members sitting around a round table, including himself. For convenience, we number Hooyeon as 1 and the rest people as $2, 3, \cdots, N$ in clockwise order. Hooyeon crafted some Poktanju with his special recipe: it is "like having your brains smashed out by a slice of lemon wrapped around a large golden brick" (excerpted from a Sci-Fi novel), a shot of which makes the person drinking it drunk so fast.

Hooyeon is currently holding a very large bottle of this Poktanju. He is going to play a simple game using the bottle. Starting from him, anyone holding the bottle must take a glass of Poktanju (and she becomes drunk immediately). Then, she throws a fair coin; if it is a head, she sends the bottle to the person on her left, but otherwise, to the person sitting on her right side. The game continues until the Sun rises.

However, Wonha (a.k.a. Being) does not like alcohols so much, especially Poktanjus. If he drinks any Poktanju, it is guaranteed that he suffers from a serious hangover. In any case, Hooyeon is quite happy about the party, so Wonha does not want to spoil Hooyeon's party. Wonha thought that if everyone except him becomes drunk, then he can cheat by drinking water instead of the Poktanju Hooyeon crafted. So Wonha wants to know the probability that everyone but him becomes drunk.

Write a program to calculate the probability.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The one and only line of each test case will contain two integers the number of people N $(2 \leq N \leq 20,000)$ and Wonha's position K $(2 \leq K \leq N)$.

Output

Print exactly one line for each test case. The line should contain the probability that Wonha is the only one who didn't become drunk. The number should be rounded to eight digits after the decimal point.

Sample input and output

Standard Input	Standard Output
2 3 2 3 3	0.50000000 0.50000000

Problem J. Lecture Note

Professor Lew teaches Algorithm course in Sonyusi University. It is his second year as a professor, so he is still spending a lot of time making lecture notes. Since he has decided to teach some tough geometry algorithms next year, he is busy making new part of the lecture note.

While making the note, he realized that he needs to draw some figures for the lecture note, as it is almost impossible to understand complicated geometry algorithms without figures. But the problem is that he is very poor at drawing, just like any other computer scientist. However, most of the figures just consist of points and vectors(lines), and it is rather easy to draw them with a computer program. Therefore, professor Lew decided to make some geometry library codes and a simple verifier program.

The program handles a rectangle with width W and height H, and some particles on the left side of the rectangle. The 2D coordinate of the lower left corner of the rectangle is $(0, 0)$, and upper right corner is (W, H). N particles are aligned on the left side of the rectangle at some particular point, and begin to move at the same time. All of these particles are moving toward the right side with some certain direction.

Each particle has its own directional vector, which indicates the moving direction. The first figure of the lecture note deals with a case when all of these particles are moving at the same $x-$directional speed. In this case, a particle at (x, y) with a directional vector (a, b) will move to $(x + 1, y + b/a)$ after one unit of time $(a > 0)$, and will move to $(x+t, y+t\cdot(b/a))$ after t units of time. When M particles with directional vectors (a_1, b_1), (a_2, b_2), $\cdots$, (a_M, b_M) meet at the same point and same time, they are merged to one point with directional vector $(M, b_1/a_1 + b_2/a_2 + \cdots + b_M/a_M)$. Moreover, when more than two particles meet at the same point and in the same time, all of these particles and directional vectors must be handled(merged) together. When a particle meets the lower or upper side of the rectangle, then it disappears and is ignored from further considerations.

The particles, if any, will reach the right side of the rectangle after W units of time. You are to find where these particles will be. Write a program that calculates the y coordinates.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

For each test case, the first line will contain three integers N $(1 \le N \le 30)$, W $(1 \le W \le 10^9)$ and H $(1 \le H \le 10^9)$, where N denotes the number of the particles. In the next N lines, three integers y $(0 < y < H)$, a $(0 < a \le 10^9)$, and b $(-10^9 \le b \le 10^9)$ are given, which means that there is a particle at $(0, y)$ with directional vector (a, b).

Output

For each test case, an integer K should be printed in the first line. In next K lines, print a real number s, rounded to four digits after the decimal point. This means that a particle meets the right side of the rectangle at a point (W, s). Note that two cases when $s = 0$ and $s = H$, before rounding, should not be printed. Also note that if two different points become equal after rounding, both of them should be printed. If two or more vectors meets at the same point of the right side, then print it once. The coordinates should be in the ascending order, and print **exactly** four digits after the decimal point.

Sample input and output

Standard Input	Standard Output
5	1
1 1 5	1.6667
1 3 2	2
6 1000000000 7	2.5000
1 10000 -1000000000	4.5000
5 211 -211	0
4 10 10	1
2 1000000000 1000000000	5.0000
3 1234 -1234	1
6 318 1000000000	3.0000
2 22 3	
2 22 1	
1 22 -1	
3 8 7	
4 1 0	
1 2 3	
6 3 -3	
5 1 6	
1 1 2	
2 1 1	
3 1 0	
4 1 -1	
5 1 -2	