Intra-IUT Junior Programming Contest 2024

https://toph.co/c/intra-iut-junior-2024

Schedule

The contest will run for **3h0m0s**.

The standings will be frozen for the last **45m0s** of the contest.

Authors

The authors of this contest are Brownbear, chroot, ir.rafio, ishtupeed, lelbaba, Reewnat.122, Ssshanto, Talha76, and user.1872.

Rules

This contest is formatted as per the official rules of ICPC Regional Programming Contests.

You can use C++17 GCC 13.2, C++20 Clang 16.0, C++20 GCC 13.2, C++23 GCC 13.2, C11 GCC 13.2, C17 GCC 13.2, C23 GCC 13.2, Java 1.8, Kotlin 1.1, PyPy 7.1 (3.6), and Python 3.12 in this contest.

Be fair, be honest. Plagiarism will result in disqualification. Judges' decisions will be final.

Notes

There are 9 challenges in this contest.

Please make sure this booklet contains all of the pages.

If you find any discrepencies between the printed copy and the problem statements in Toph Arena, please rely on the later.

A. Colorful Socks

Limits 1s, 512 MB

Abid has a beautiful collection of socks with colors 1 to n. There are a_i socks of color i in his drawer.

While preparing for a party, Abid finds himself in a tough spot when he realizes that his brother swiped some socks, taking b_i socks of color i from his drawer.

Suddenly, the lights go out and Abid can not tell the colors apart. He wants to grab some socks and later put on a pair of socks with the same color.

Help Abid by telling him the minimum number of socks he needs to grab.

Input

The input contains three lines.

The first line contains an integer $n(1 \le n \le 2 \times 10^5)$ --- the number of different colors of socks Abid had.

The second line contains n space-separated integers a_1, a_2, ..., a_n $(1 \le a_i \le 10^9)$ --- the initial number of socks of each color in Abid's drawer before his brother took some. a_i represents the initial number of socks of color i.

The third line contains n space-separated integers b_1, b_2, ..., b_n $(0 \le b_i \le a_i)$ --- the number of socks of each color that Abid's brother took from the drawer. b_i represents the number of socks of color i that Abid’s brother took.

Output

Output a single integer in a line --- the minimum number of socks Abid must grab to guarantee that he finds at least 2 socks of the same color.

If it is not possible to find a pair of socks of the same color, output -1.

Samples

Input	Output
3 4 4 4 1 1 1	4

Input	Output
In this case, if Abid grabs 4 socks, it can be proven that he will find at least 2 socks of the same color.	

Input	Output
1 4 0	2
In this case, Abid's brother did not take anything, and all the socks in the drawer have the same color. So Abid can grab any 2 socks.	

Input	Output
2 5 10 4 9	-1
In this case, all the socks remaining in the drawer have different colors.	

B. Darhi Palla

Limits 1s, 512 MB

The weight balance is one of the oldest mechanical devices used by humans. Its origins can be traced back to ancient civilizations such as the Egyptians, who used simple balances to measure commodities like grain.

A weight balance typically consists of a beam supported at its center, with two pans suspended from either end. The object to be weighed is placed on one pan, while standard weight stones are placed on the other until equilibrium is reached, indicating that the weights are equal.

Stones can also be placed on the same pan as the object. If you place the object on the right pan, place stones totaling weight X on the left pan, stones totaling weight Y on the right pan and the balance reaches equilibrium, then the weight of the object is $X - Y$.

For example, if the balance reaches equilibrium while having a stone of weight 9 on the left pan and a stone of weight 3 and an object on the right pan, then the weight of the object is $9 - 3 = 6$.

You want to check if the weight of an object is exactly W. You have placed it on the right pan and need to place stones on both pans to reach equilibrium.

You have 999999937 stones. The first stone has a weight of 1, and each subsequent stone is 3 times heavier than the previous one. So, for example, the weights of the first five stones are 1, 3, 9, 27, and 81.

You have to find two integers X and Y such that you can place stones totaling weight X and Y on the left and right pan respectively and $X - Y = W$.

Check the sample test cases and see the constructions for a better understanding.

Assume that all the events are taking place on Earth, so weight and mass are proportional.

Input

The first line of the input contains a single positive integer t $(1 \leq t \leq 10^5)$ --- the number of test cases. Then t test cases follow.

Each test case contains only one integer in a line W $(1 \le W \le 2 \times 10^9)$ --- the expected weight.

Output

For each test case, output two space-separated integers in a line X and Y --- the total weight of stones on the left and right pan respectively.

Since the object is on the right pan, X must be greater than Y.

If there are multiple correct combinations, you may output any of them.

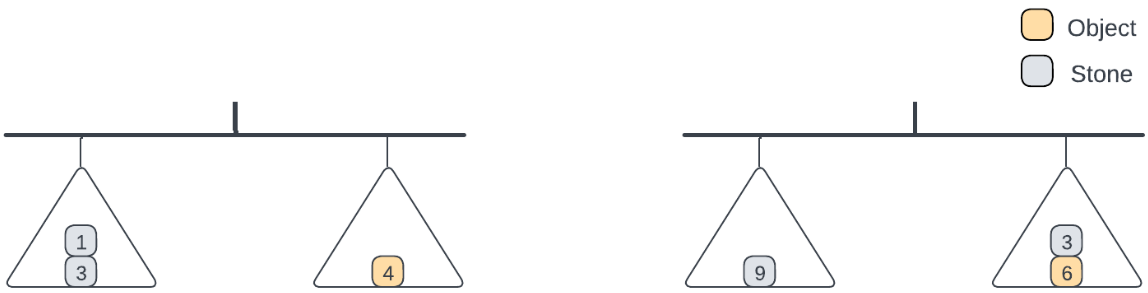
If there is no correct combination, output "-1 -1".

Samples

Input	Output
2 4 6	4 0 9 3

Input	Output
1 100	109 9

The construction of the first test case is illustrated in the following image:



C. World Peace!

Limits 1s, 256 MB

Bondman has been given a top-secret mission.

Eden Academy is celebrating its orientation party. All the students are enjoying themselves before their class officially begins. The school has received some gift boxes for this occasion. The gift boxes are lined up on a table from left to right. Each gift box has some number of toys. The principal will try to distribute them evenly among the students. But there might be toys left over that cannot be distributed, which will surely cause a war to break out among the students and ruin the peace of the world!

Bondman must stop that no matter what! To accomplish this, he can remove some (possibly 0) gift boxes from the left and some from the right side of the table. But he cannot remove any from the middle, as it will make it obvious that some boxes are missing! He also cannot remove all the boxes, since the school has already made an announcement about the gift boxes. In order to avoid raising suspicion, he wants to remove the minimum number of boxes from the table.

Now, Bondman wants to know the minimum number of boxes he can remove to fulfill his mission.

Input

The first line of the input contains a single positive integer T $(1 \le T \le 10^4)$ — the number of test cases. Then T test cases follow.

The first line of each test case contains two space-separated positive integers n $(1 \le n \le 10^5)$ and m $(1 \le m \le 10^9)$— the number of gift boxes on the table and the number of students.

The second line of each input contains n space-separated positive integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10^9)$—- where a_i is the number of toys in the i-th gift-box from the left.

The sum of n over all test cases will not exceed 10^5.

Output

For each test case, output one integer in a line. If Bondman cannot succeed in his mission, you must print -1. Otherwise, print the minimum number of gift boxes Bondman has to remove.

Samples

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

In the first test case, if we keep all the gift-boxes, there will be a total of 13 toys, which cannot be divided equally among 4 people. If we remove the first gift-box, the total will be 12, which can be divided among 4 people.

In the second test case, the total toys in all the gift-boxes is 5, which cannot be divided equally among 6 people without removing all the gift-boxes.

In the third test case, removing the left-most and the right-most gift-box is the only possible way to fulfill the mission.

In the fourth test case, it can be shown that removing the last two gift-boxes is optimal.

Input	Output
5 2 6 16 21 4 5 12 30 7 14 6 2 12 23 9 3 20 6 6 19 15 5 30 24 28 18 3 5 17 18 29	-1 3 2 -1 1

D. Kingslayer

Limits 1s, 512 MB

In the nation of Britannia, the rules of Chess are simpler than what we know. Here, you do not win by checkmating the opponent, but instead by capturing the opponent's king.

You are Schneizel el Britannia, the White Prince of the Holy Britannian Empire, and you are playing with the white pieces. Unlike your brother Lelouch vi Britannia, you are a Cold Blooded Strategist. You are not afraid to go for the kill when your opponent mistakenly leaves the king open for capture.

The pieces in Britannian Chess move similarly to regular chess, namely:

1. Pawn (P or p): Can only move straight ahead, one square at a time. However, it captures pieces diagonally (only forward).
2. Knight (N or n): The only piece that can jump over other pieces. The knight movement can be viewed as an "L", as clarified in the example below (forward or backward).
3. Bishop (B or b): Can move any number of squares diagonally (forward or backward).

4. Rook (R or r): Can move any number of squares vertically or horizontally (forward or backward).

5. Queen (Q or q): Can move any number of squares diagonally, horizontally, or vertically (forward or backward).

6. King (K or k): Can move one square at a time diagonally, horizontally, or vertically (forward or backward).

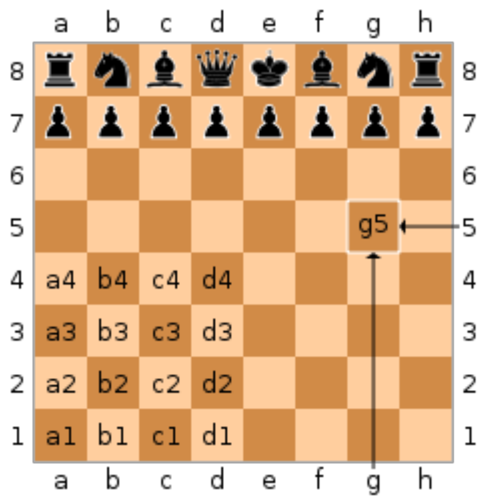
Examples:

```
Pawn        Rook        Bishop      Queen       King        Knight
........    ...*....    .......*    ...*...*    ........    ........
........    ...*....    *.....*.    *..*..*.    ........    ........
........    ...*....    .*...*..    .*.*.*..    ........    ..*.*...
..*.*...    ...*....    ..*.*...    ..***...    ..***...    .*...*..
...P....    ***R****    ...B....    ***Q****    ..*K*...    ...N....
........    ...*....    ..*.*...    ..***...    ..***...    .*...*..
........    ...*....    .*...*..    .*.*.*..    ........    ..*.*...
........    ...*....    *.....*.    *..*..*.    ........    ........
```

'*' indicates where the piece can capture another piece.

Note: Britannian Chess is simple, and all possible moves are mentioned above. Complex moves in regular chess such as castling, pawn promotions, and *en passant* do not exist in Britannian Chess.

Chessboard positions are defined in algebraic notation, as shown here:



In this game, consider that your opponent has made his move and it is your (white's) turn to move. Given the board formation, determine whether you can win the game, ie. capture the opponent's king, in a single move. If it is possible, you need to print the lexicographically smallest position from which you can capture the opponent's king.

A string **a** is lexicographically smaller than string **b** (of the same length) if, in the first position where **a** and **b** differ, string a has a letter that appears earlier in the alphabet than the corresponding letter in b. For example, position "b8" is lexicographically smaller than "c1" because 'b' comes before 'c' in dictionary order.

Input

The first line of the input contains a single positive integer T $(1 \le T \le 10^4)$ — the number of test cases. Then T test cases follow.

Each test case contains 8 lines of 8 characters each describing the board formation. An empty line separates two adjacent test cases.

White pieces will be represented by uppercase letters whereas black pieces will be represented by lowercase letters, as defined above. White (you) always starts the game from the bottom of the board and black always starts the game from the top of the board.

. denotes an empty square.

It is guaranteed that exactly one black king k and exactly one white king K exist on the board.

Output

For each test case:

If it is possible for you to win in one move, output *"YES"* (without quotes) on the first line. On the second line, print the lexicographically smallest position from which you can capture the opponent's king.

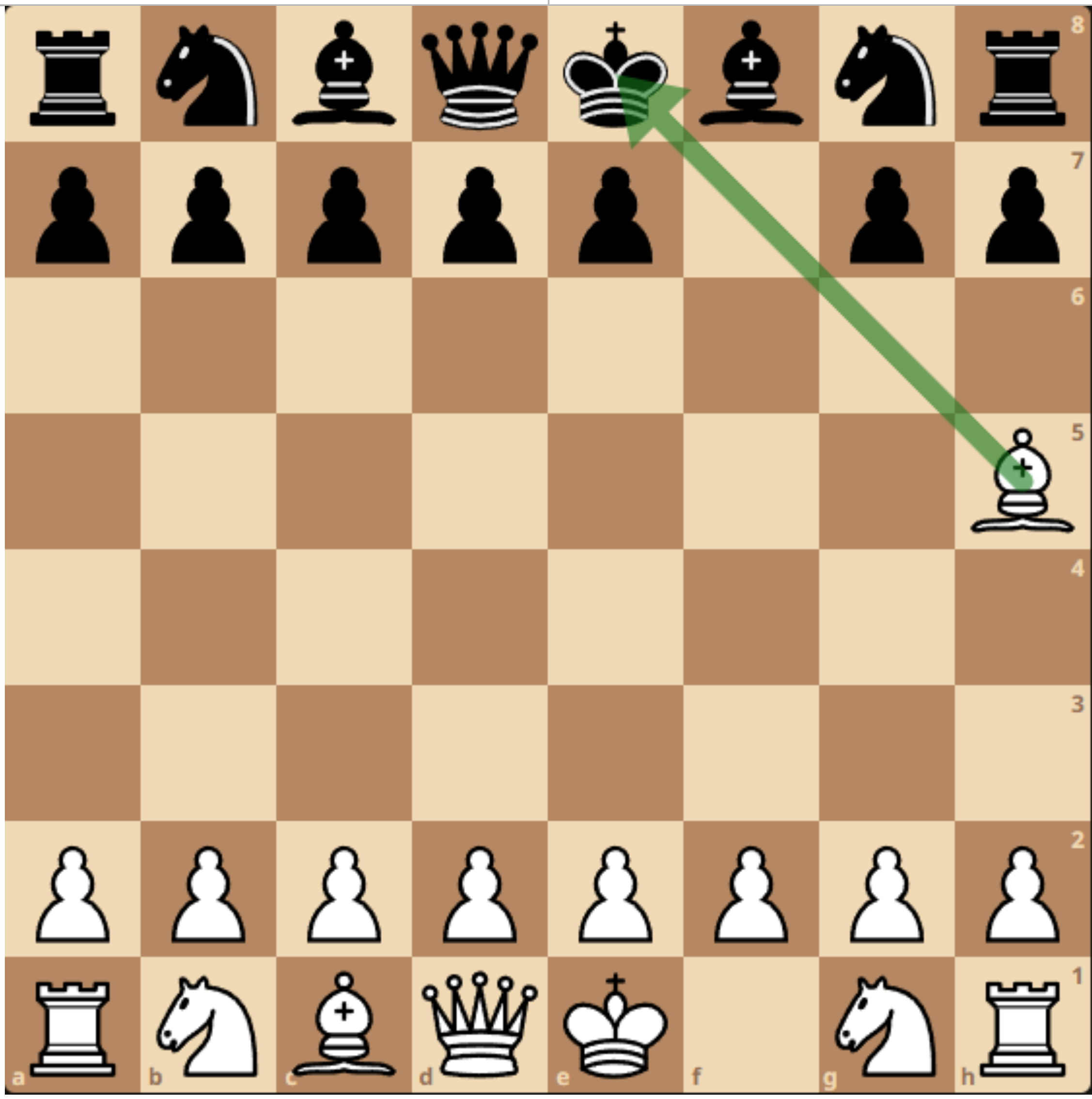
If it is not possible to win in one move, output *"NO"*.

The checker is case-insensitive. The string *"YES"*, *"YeS"*, *"yes"* and *"Yes"* are considered the same.

Samples

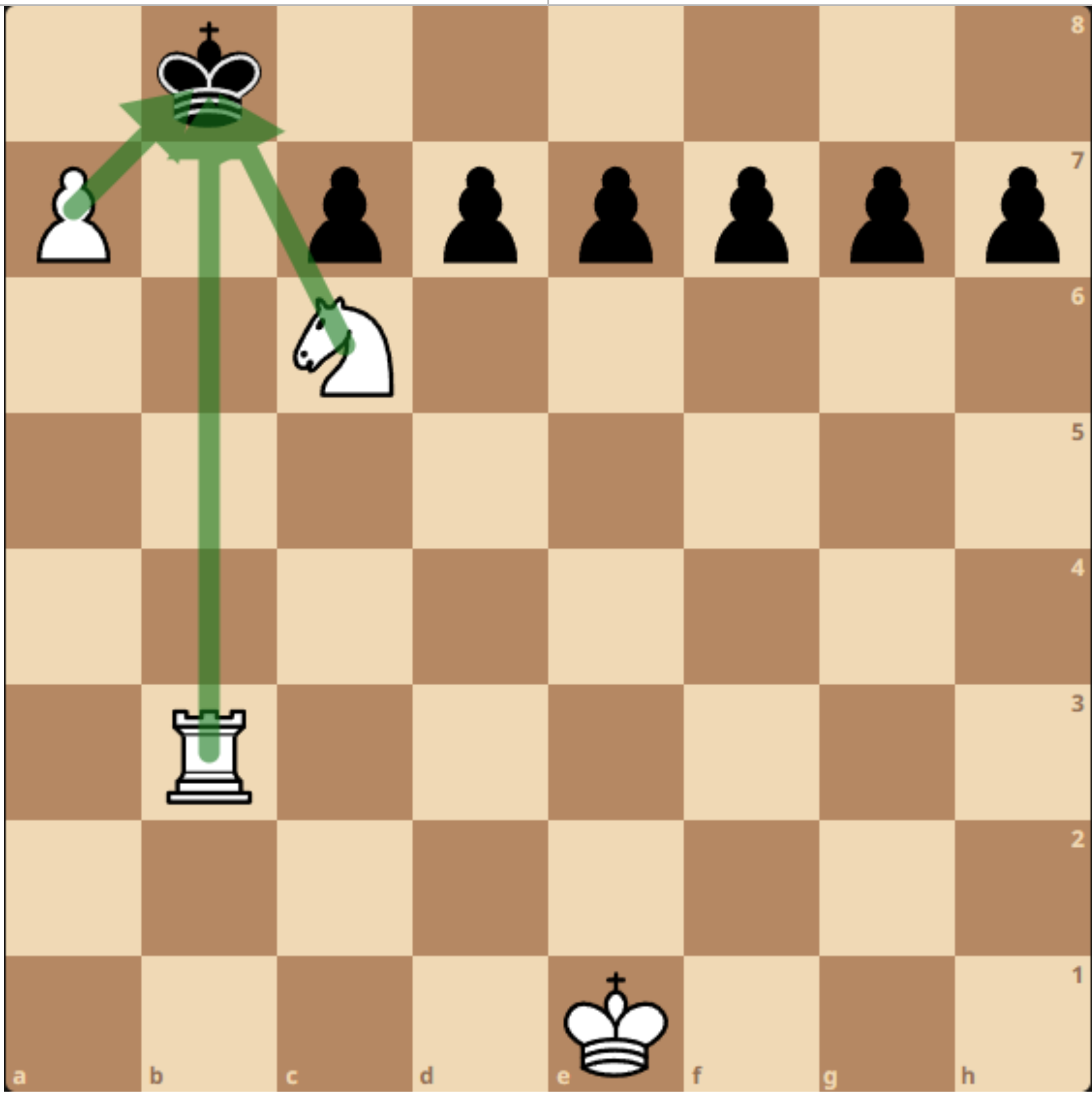
Input	Output
5 rnbqkbnr ppppp.pp ....B	YES h5 YES a7 NO YES

Input	Output
.... PPPPPPPP RNBQKBNR .k...... P.pppppp ..N..... .... .R...... ....K... ....P.... ..k..... ....K ....k.... ..K..... ....k... ...n.... ....n... B....... ....R... .K......	c4 NO
Case 1: The white bishop at position $h5$ can capture the black king.	

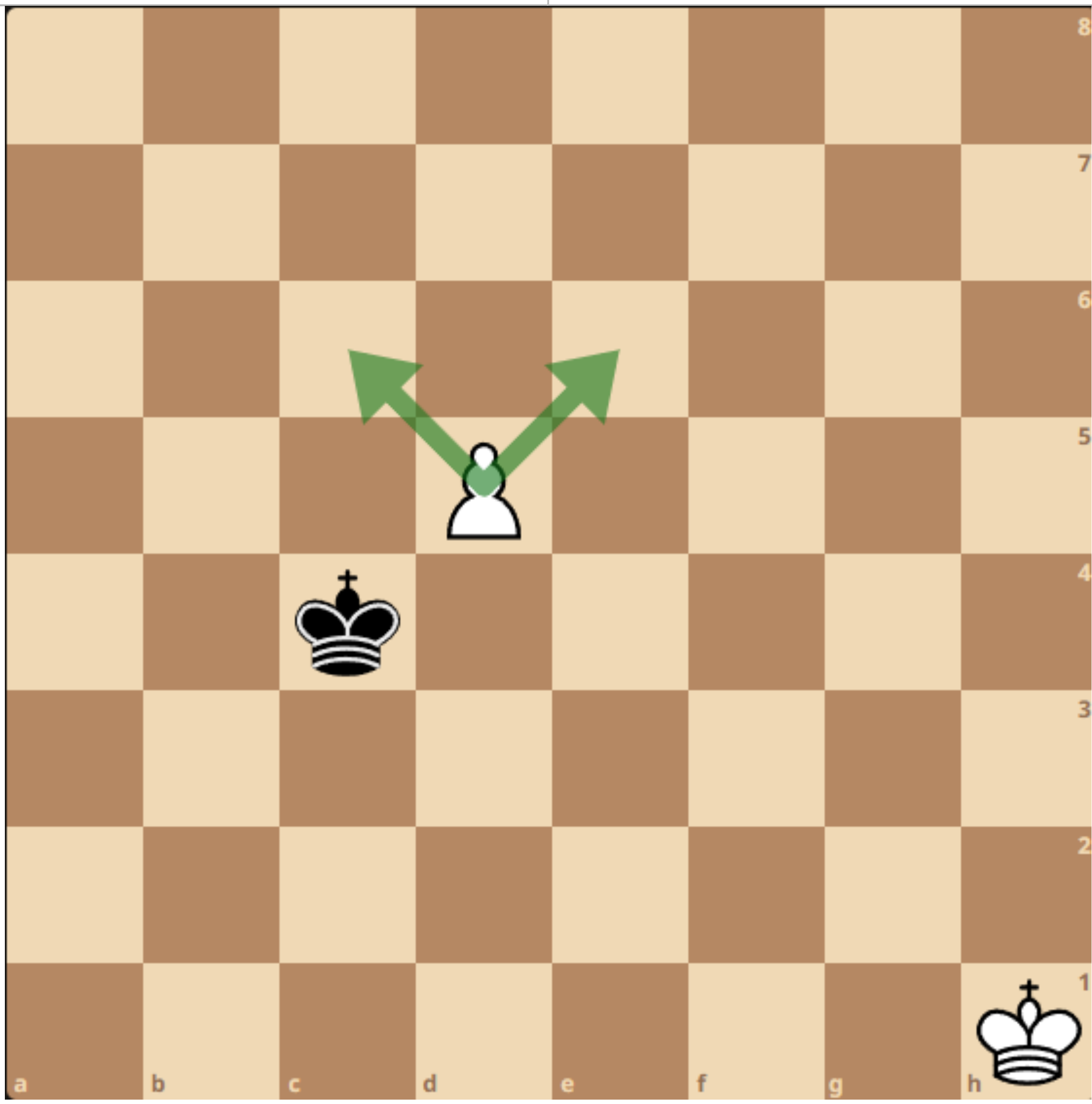
<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td colspan="2">

Case 2: The white pawn at $a7$, white rook at $b3$, and white knight at $c6$ can capture the black king. Among all three positions, $a7$ is the smallest lexicographically.</td></tr>
</table>

Input	Output

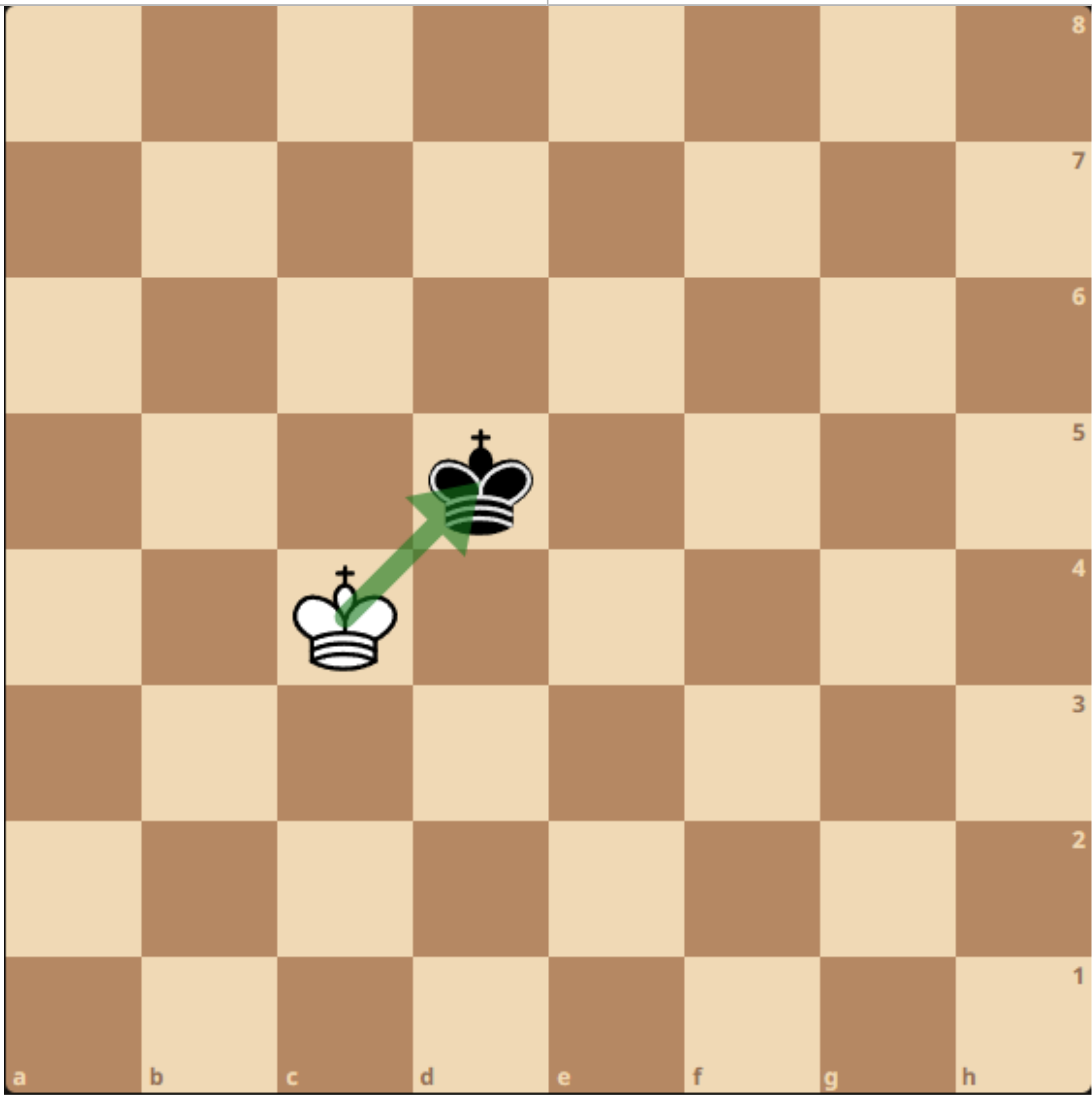


Case 3: The white pawn at $d5$ can only capture forward. Hence it cannot capture the black king behind it.

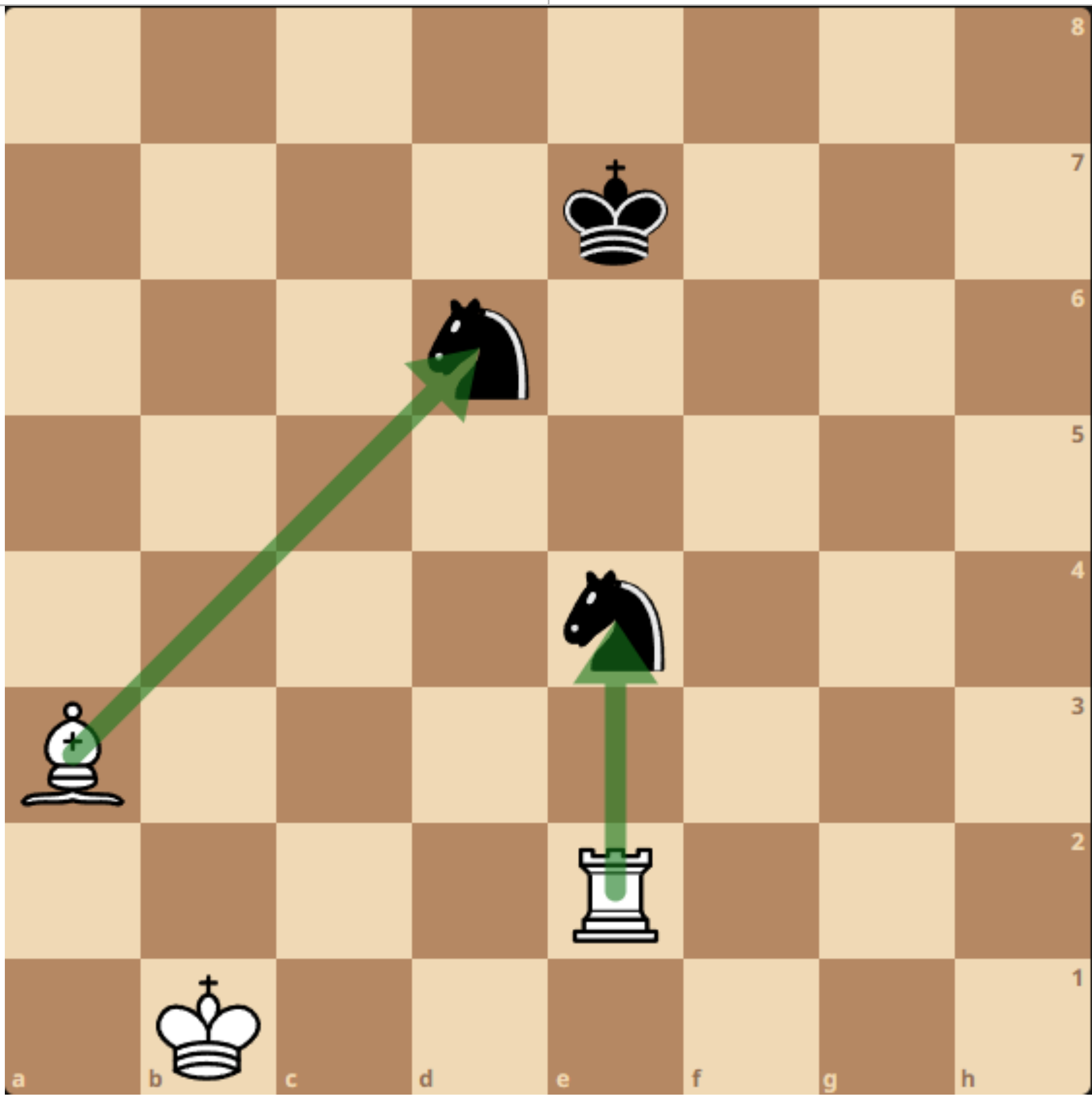
<table>
<tr><th><u>Input</u></th><th><u>Output</u></th></tr>
<tr><td colspan="2">


Case 4: The white king at $c4$ can capture the black king.
</td></tr>
</table>

Input	Output



Case 5: The black knights are protecting their king, so the black king can’t be captured in a single move.

Input	Output
	

E. Monks' Game of Cards

Limits 1s, 512 MB

Moudud, Shantanu, and Sumit are playing a game with a deck of n cards. In this deck, each card is marked with a number written on it. Sumit will shuffle the deck of cards, and then Shantanu will cut the deck at a certain position. Moudud will have to guess the number written on the card on top of the deck after the shuffle and cut.

To prevent the game from becoming a boring game of chance, Sumit will tell Moudud his shuffle order. Before every round, Sumit will tell Moudud that he will shuffle the deck k times and Shantanu will tell Moudud his cut position p. After every round, the deck will be rearranged to its original order.

Now Moudud wants your help to win. Can you help him find the number written on the card on top of the deck?

A deck of cards is represented by an array a of size n, where a_i denotes the number written on the i^{th} card. The position of the cards is numbered from top to bottom and a_1 is the number written on the card at the top. The i^{th} card can also be referred to as the card at position i

A shuffle order is a permutation of size n (array of size n consisting of all integers from 1 to n), denoted as s, where each element s_i represents the original position of the card that should occupy the position i after shuffling. Applying a shuffle with this order rearranges the deck such that the card at position s_i before the shuffle moves to position i. Formally, $a'_i = a_{s_i}$, where a_i is the number written on the i^{th} card before the shuffle, a'_i is the number written on the i^{th} card after the shuffle and s is the shuffle order.

In the cut operation at a specified position p, the deck is split at the position p. The deck is divided into two parts: the first part consists of cards from position 1 to position p, and the second part consists of cards from position $p+1$ to the end. After cutting, the two parts are swapped (the second part is brought before the first), resulting in a new arrangement of cards. Formally, cutting a deck a_1, a_2, ..., a_n transforms it into a_{p+1}, a_{p+2}, ..., a_n, a_1, a_2, ..., a_p.

Input

The first line of the input contains a single positive integer t $(1 \leq t \leq 10^5)$ --- the number of test cases. Then t test cases follow.

The first line of every test case contains two space-separated integers n $(2 \le n \le 2 \times 10^5)$ and r $(1 \le r \le 10^5)$ --- the number of cards in the deck and the number of rounds they will play the game.

The second line of every test case contains n space-separated integers a_1, a_2, ..., a_n --- the array representing the deck of cards, where a_i $(1 \le a_i \le 10^9)$ denotes the number written on the i^{th} card.

The third line of every test case contains n space-separated integers s_1, s_2, ..., s_n that contains all integers from 1 to n --- Sumit's shuffle order.

The last r lines of every test case contain two space-separated integers k $(0 \le k \le 10^{18})$ and p $(1 \le p < n)$ --- the number of times Sumit will shuffle the deck and the position where Shantanu will cut.

The sum of $n \times r$ will not exceed 2×10^5 over all test cases.

Output

For each test case, output r lines with one integer per line, representing the number written on the card on top of the deck in each round.

Samples

<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td><pre>3
5 3
10 20 30 40 50
3 5 4 1 2
2 4
0 1
50 2
7 7
240 240 240 240 240 240 240
5 6 3 1 4 7 2
4724907009358229 6
3 5
7 3
1 3
2 4
6 6
4 3
2 1
3 5
2 1
1 1</pre></td><td><pre>50
20
10
240
240
240
240
240
240
240
3</pre></td></tr>
<tr><td colspan="2">In the first round of the first test case, the initial deck is $10, 20, 30, 40, 50$.</td></tr>
</table>

Input	Output
After the first shuffle, it becomes $30, 50, 40, 10, 20$. After the second shuffle, it becomes $40, 20, 10, 30, 50$. After the cut, it becomes $50, 40, 20, 10, 30$.	

Integers larger than $2^{32} - 1$ will not fit inside an **int** variable. For these values, use the data type **long long**.

F. Moniter Goja, Gojar Monit

Limits 1s, 512 MB

Given a natural number k such that $k > 1$, you have to find two natural numbers x and y such that it satisfies this equation:

$$x^y + y^x = k$$

Input

The first line of the input contains a single positive integer t $(1 \leq t \leq 10^5)$ — the number of test cases. Then t test cases follow.

Each test case contains only one integer in a line k $(2 \leq k \leq 10^{18})$.

Output

For each test case, output two space-separated natural numbers in a line — the values of x and y that satisfy the equation.

Recall that a natural number is a positive integer. So both x and y must be greater than 0.

If there are multiple correct combinations, you may output any of them.

It is guaranteed that at least one solution exists.

Samples

Input	Output
2 17 6250	2 3 5 5

$2^3 + 3^2$ = $8 + 9$ = 17.

$5^5 + 5^5$ = $3125 + 3125$ = 6250.

Integers larger than $2^{32} - 1$ will not fit inside an **int** variable. For these values, use the data type **long long**.

G. MDP

Limits 1s, 512 MB

Consider that a famous Turkish architect, Mustafa Duru Poyraz (MDP) has been assigned to design the campus for the Institute of Urban Transformation (IUT), a subsidiary organ of the Overseas Investment Council (OIC). The OIC has the following member states:

- Republic of Firelands
- Hashemite Realm of Petra
- Afghan Islamic Federation
- Republic of Illyria
- Emirati Federation
- Indonesian Archipelago
- Republic of Samarkand
- Ugandan Savannah Union
- Persian Union
- Paksitan Federation
- Kingdom of Pearls
- Bruneian Sultanate
- Bengal Peoples' Union
- Republic of Dahomey
- Burkina Greenheart
- Republic of Tadzhik
- Anatolian Union
- Turkmensky Dominion
- Republic of Sahel
- Togo Union

- Republic of Carthage
- People's Maghreb Union
- Djiboutian Horn Union
- Saudi Sand Kingdom
- Senegambian Republic
- Sudanic Republic
- Levantine Arab Republic
- Suriname Rainforest Union
- Sierra Leonean Freedom Republic
- Somali Coastal Union
- Iraqi Tigris
- Omani Sands Sultanate
- Gabonese Coastal Republic
- Gambian Riverlands Republic
- Guyanese Rainforest Republic
- Guinean Coastal Republic
- Bissau Coastal Republic
- Palestinian Levantine State
- Comorian Archipelago Union
- Kyrgyz Mountain Republic
- Qatari Peninsula State
- Kazakh Steppe Republic
- Cameroonian Savannah Republic
- Ivorian Coast Republic
- Kuwaiti Desert State

- Lebanese Cedar Republic
- Libyan Desert Nation
- Maldivian Atoll Republic
- Malian Sahel Republic
- Malayan Peninsula Union
- Egyptian Nile Republic
- Moroccan Maghreb Kingdom
- Mauritanian Sahara Republic
- Mozambican Coastal Republic
- Nigerien Sahel Republic
- Nigerian Savanna Republic
- Yemeni Arabian Republic
- Levantine Arab State
- Spice Islands

The campus of IUT will include one flag stand for each member state. Additionally, there will be three (3) separate flag stands in front of the Culinary Delights Station (CDS) of IUT. Since MDP is busy designing the campus, he has asked you to count the number of flag stands that need to be placed.

Input

There is no input to the problem.

Output

A single integer, indicating the number of flag stands in IUT.

H. It's Time to Duel!

Limits 1s, 512 MB

Yugi and Kaiba are engaged in a heated duel of cards. Yugi has brought n_1 cards and Kaiba has brought n_2 cards. Each card has a number written on it. In this duel, a player can use one or two cards from his respective deck and his score is determined by the product of the numbers on the cards he has used. The player with the higher score wins the duel.

Given the decks of Yugi and Kaiba, determine which player will emerge victorious in this duel. It is guaranteed that both Yugi and Kaiba, fueled by their bitter rivalry, will always play optimally.

Input

The first line of the input contains a single positive integer t $(1 \le t \le 10^4)$ — the number of test cases. Then t test cases follow.

Each test case contains three lines.

The first line of a test case contains two space-separated integer n_1 $(2 \le n_1 \le 2 \times 10^5)$ and n_2 $(2 \le n_2 \le 2 \times 10^5)$ — the number of cards in Yugi and Kaiba's deck.

The second line contains n_1 space-separated integers a_1, a_2, ..., a_{n_1} $(-10^9 \le a_i \le 10^9)$ — the number written on every card of Yugi's deck.

The third line contains n_2 space-separated integers b_1, b_2, ..., b_{n_2} $(-10^9 \le b_i \le 10^9)$ — the number written on every card of Kaiba's deck.

It is guaranteed that the sum of $n_1 + n_2$ over all test cases does not exceed 2×10^5.

Output

For every test case, output "Yugi" (without the quotes) if Yugi will win the duel, "Kaiba" if Kaiba will win the duel, and "Tie" if the duel will end in a tie.

The checker is case-insensitive. That means, for example, the string *"Yugi", "yUgI"* and *"YUGI"* are considered the same.

Samples

<table>
<tr><th><u>Input</u></th><th><u>Output</u></th></tr>
<tr><td><pre>4
4 3
0 8 0 1
-2 3 3
4 4
-5 6 4 -5
5 4 -5 3
2 3
-3 -5
-15 1 15
2 2
-3 8
4 4</pre></td><td><pre>Kaiba
Yugi
Tie
Kaiba</pre></td></tr>
<tr><td colspan="2">

In the first case, if Kaiba chooses his second and third card, his score is $3 \times 3 = 9$. Yugi can't beat or equal this score.

In the second case, if Yugi chooses his second and third card, he has a score of 24. Kaiba can't beat or equal this score.

In the third case, if Yugi plays just his first card (-3)or just his second card(-5), Kaiba will play his third card(15) to win. So Yugi must play both cards $(-3 \times -5 = 15)$. No matter how Kaiba plays, Kaiba's score cannot increase from 15. So the duel will end in a tie.

In the fourth case, Kaiba will play both his cards $(4 \times 4 = 16)$. No matter what Yugi plays, Yugi will lose.

</td></tr>
</table>

<table>
<tr><th><u>Input</u></th><th><u>Output</u></th></tr>
<tr><td><pre>3
5 5
-5 4 6 0 -7
-2 -15 0 3 10
4 5
2 3 -1 -5
-1 -2 -1 0 7
3 3
-3 -3 5
9 -4 -2</pre></td><td><pre>Yugi
Kaiba
Tie</pre></td></tr>
</table>

I. Forever Matrix

Limits 500ms, 256 MB

Anabia has challenged Abrar to a game of wits. She has given Abrar a problem - and despite his best efforts, Abrar is unable to solve it. Out of desperation he seeks the greatest mathematician, you, for help. The problem goes as follows:

Anabia defined a new type of matrix, named **Forever Matrix**, as below:

$$M_0 = \begin{bmatrix} 1 \end{bmatrix}$$

$$M_n = \begin{bmatrix} M_{n-1} & a \times M_{n-1} \\ b \times M_{n-1} & c \times M_{n-1} \end{bmatrix};\ n > 0$$

For, example, if $a = 2$, $b = 3$ & $c = 5$, then

$$M_0 = \begin{bmatrix} 1 \end{bmatrix}$$

$$M_1 = \begin{bmatrix} M_0 & 2 \times M_0 \\ 3 \times M_0 & 5 \times M_0 \end{bmatrix} = \left[\begin{array}{c|c} 1 & 2 \\ \hline 3 & 5 \end{array}\right]$$

$$M_2 = \begin{bmatrix} M_1 & 2 \times M_1 \\ 3 \times M_1 & 5 \times M_1 \end{bmatrix} = \left[\begin{array}{cc|cc} 1 & 2 & 2 & 4 \\ 3 & 5 & 6 & 10 \\ \hline 3 & 6 & 5 & 10 \\ 9 & 15 & 15 & 25 \end{array}\right]$$

$$M_3 = \begin{bmatrix} M_2 & 2 \times M_2 \\ 3 \times M_2 & 5 \times M_2 \end{bmatrix} = \left[\begin{array}{cccc|cccc} 1 & 2 & 2 & 4 & 2 & 4 & 4 & 8 \\ 3 & 5 & 6 & 10 & 6 & 10 & 12 & 20 \\ 3 & 6 & 5 & 10 & 6 & 12 & 10 & 20 \\ 9 & 15 & 15 & 25 & 18 & 30 & 30 & 50 \\ \hline 3 & 6 & 6 & 12 & 5 & 10 & 10 & 20 \\ 9 & 15 & 18 & 30 & 15 & 25 & 30 & 50 \\ 9 & 18 & 15 & 30 & 15 & 30 & 25 & 50 \\ 27 & 45 & 45 & 75 & 45 & 75 & 75 & 125 \end{array}\right]$$

$$\vdots$$

This goes on forever and the resulting matrix is infinite in height and width.

You are given the values of a, b, and c and the position of a cell (x, y) where x and y denote the row number and the column number of the **Forever Matrix**, respectively. The rows are numbered from top to bottom, the columns are numbered from left to right and the position of the top-left cell is $(0, 0)$.

Your task is to output the value at the given cell modulo $10^9 + 7$.

Input

The first line contains a positive integer t $(1 \le t \le 3 \times 10^5)$ — the number of test cases. Then t test cases follow.

The first line of each test case contains three space-separated positive integers a, b and c $(0 < a, b, c \le 10^9)$.

The second line of the test case contains two space-separated integers x and y $(0 \le x, y \le 10^9)$.

Output

For each test case, print a single integer — the answer modulo $10^9 + 7$.

Samples

Input	Output
2 1 1 1 2 2 2 3 5 7 1	1 45

Input	Output
1 20 30 50 4 5	1000

Test set is huge. Use fast I/O.