

Problem A. Eye Color

Time limit: 1 second
Memory limit: 512 megabytes

In a distant land inhabited by magical creatures, geneticists have discovered a unique phenomenon: the eye color of offspring directly depends on the eye color of their parents. In this world, there are only three eye colors: brown, green, and blue.

Geneticists have developed several rules that determine what eye color a child can have:

- If at least one parent has brown eyes, then the child will definitely have brown eyes.
- Otherwise, if a parent has green eyes, then the child will also have green eyes.
- If both parents have the same eye color, then the child will have the same color.

The geneticists have an ancient book with statistics on the eye colors of parents and offspring. However, it may contain incorrect entries or the symbol '.', which indicates that the eye color was not recorded and can take any of the three values.

Your task is to help the geneticists verify the correctness of the records regarding eye colors. You are given the eye colors of both parents and their offspring. You need to determine all possible correct combinations of eye colors for the parents and the offspring in lexicographic order. If such combinations are impossible, output the word "Incorrect".

Input

You are given three space-separated eye colors: two parents and their offspring. The eye color can take the values brown, green, blue, or '.'.

Output

Output all possible correct combinations of eye colors for the parents and the offspring, separated by spaces, in lexicographic order, with each combination on a separate line. Or the word "Incorrect" if the combination is unattainable.

Examples

standard input	standard output
blue blue blue	blue blue blue
blue brown .	blue brown brown
green . .	green blue green green brown brown green green green
blue blue brown	Incorrect

Problem B. Big World Politics

Time limit: 1.5 seconds
Memory limit: 512 megabytes

You are given a map of the world, represented as a rectangle of size $n \times m$. Each cell of the map belongs to one of k countries. The set of cells of each country is connected by sides—this means that from any cell, you can reach any other cell of the same country by moving down, up, left, and right, without entering the territory of another country.

In these troubled times, many countries have territorial claims against each other. Country i considers all cells lying within the minimum rectangle with sides parallel to the edges of the map that contains all cells belonging to the country as its historical territories. Thus, country i has territorial claims against all countries j that have cells within the minimum rectangle of country i , except for country i itself.

As a professional analyst, your task is to determine for each country i how many countries it has territorial claims against.

Input

The first line of input contains three integers n , m , and k ($1 \leq n, m \leq 2 \cdot 10^5$, $1 \leq k \leq nm \leq 2 \cdot 10^6$) — the height of the map, the width of the map, and the number of different countries, respectively.

The following n lines contain m integers each, where the i -th line contains $a_{i1}, a_{i2}, \dots, a_{im}$ ($1 \leq a_{ij} \leq k$) — the numbers of the countries to which the cells $(i, 1), (i, 2), \dots, (i, m)$ belong, respectively.

It is guaranteed that each of the k countries has at least one cell.

It is guaranteed that the set of cells of each country is connected by sides.

Output

Output k integers, where the i -th integer is the number of countries that country i has territorial claims against.

Example

standard input	standard output
3 4 4	2 1 0 0
1 3 3 2	
1 2 2 2	
1 1 1 4	

Problem C. Puzzle

Time limit: 1 second
Memory limit: 512 megabytes

You are presented with a puzzle. Given a table of n rows and m columns filled with zeros and ones. Only one type of move is allowed: within any column, you can rearrange the elements freely, changing the order of the rows in that column in any way. The number of zeros and ones in each column remains the same.

You are allowed to make as many such moves as you want. The goal of the puzzle is to obtain as many completely identical rows as possible.

Determine the maximum number of rows that can be made completely identical using the described moves.

Input

The first line contains two integers n and m ($1 \leq n, m \leq 10^5$; $nm \leq 2 \cdot 10^5$) — the number of rows and columns.

Each of the following n lines contains m characters '0' or '1' — the elements of the initial table.

Output

Output a single integer — the maximum number of identical rows that can be obtained.

Example

standard input	standard output
3 4 0101 0010 0100	2

Problem D. New Balls

Time limit: 1 second
Memory limit: 512 megabytes

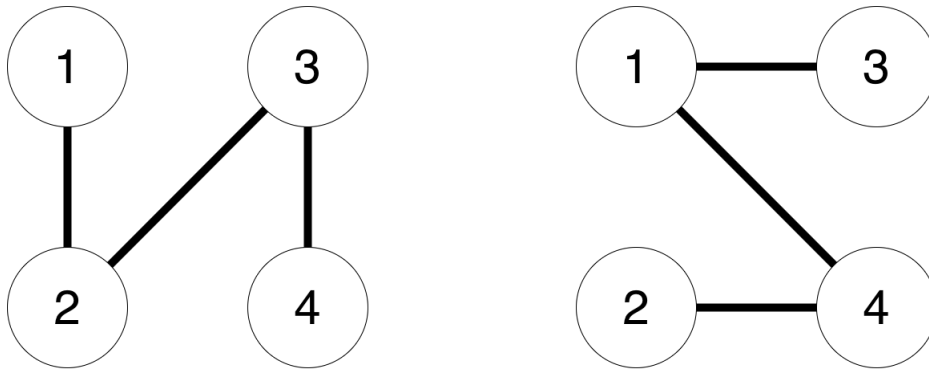
This is an interactive problem.

Usually, participants of the VKOSHP receive balls for each solved problem, but this time, for each solved problem, the team will be given a tree! Recall that a tree is a connected undirected graph without cycles. Since there are n problems in the contest, it is necessary to prepare n different trees, each containing $2n$ vertices, numbered from 1 to $2n$.

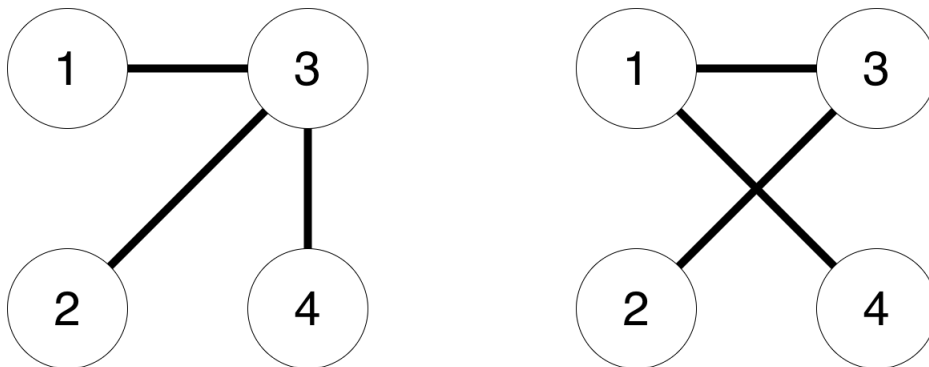
However, it turned out that trees tend to change constantly. If you start with some tree, before it is delivered to the team for the solved problem, the following operation can be applied to the tree no more than $n - 1$ times:

- Choose numbers a, b, c, d such that the edge (a, b) is in the tree, and the edges (c, d) are not in the tree;
- Then, remove the edge (a, b) from the tree and add the edge (c, d) . It is guaranteed that after this operation, the graph will remain a tree.

You are interested in distinguishing the solved problems of each team, so you need to find n trees such that they can be distinguished from each other even after the described modifications.



The image above shows an example of balls from test cases for $n = 2$.



The trees above show an example of the required trees from the test cases for your program.

The testing of this problem is organized as follows. First, you report to the jury program n trees, each containing $2n$ vertices.

After that, you will be given q trees in turn, each of which was obtained from one of your trees by applying no more than $n - 1$ of the operations described above. For each tree, you must indicate which original tree it was derived from.

Interaction Protocol

First, your program should read from standard input the numbers n and q ($1 \leq n, q \leq 100$).

Then, it should output n trees, for each tree you need to output $2n - 1$ pairs of numbers—description of the edges of the tree.

After that, the jury program will give you q times a tree obtained from operations described above, and for each query, you must output a single number x —the index of the original tree from which the jury's tree was obtained.

Example

In sample test interaction messages between the jury program and the contestant program are separated by empty lines to visualize which message is a response to which. In real interaction there will be no empty lines and you should not print any.

standard input	standard output
2 2	1 2 2 3 3 4 1 4 2 4 1 3
2 3	
1 3	
4 3	1
2 3	
1 3	
1 4	2

Note

After each action of your program, output a newline.

After each action of your program, flush the output stream.

If you use `writeln` in Pascal, `cout << ... << endl` in C++, `System.out.println` in Java, `print` in Python, `Console.WriteLine` in C#, then the output stream is flushed automatically, and you do not need to do anything additional. If you use another method of output, it is recommended to flush the output stream. Note that a newline must be output in any case.

Problem E. Array Depletion

Time limit: 1 second
Memory limit: 512 megabytes

After a long journey home, the hare Xenia decided to play a game on her bunny phone.

The level in the game is described by an array of length n and an integer x . In one move, you can take 2 adjacent elements of the array whose sum equals x and remove them. After that, the size of the array will decrease by 2. You win the level if, after a certain number of moves, the array becomes empty.

The current level seemed very difficult to her, and she even doubted whether it was possible to complete it — after all, there might be a situation where the game creators did not come up with more levels and made the last level impossible. Xenia turned to you for help — tell her if it is possible to complete this game.

Input

The first line contains two integers n, x — the size of the array and the sum of the elements to be removed ($1 \leq n \leq 3 \cdot 10^5, -10^9 \leq x \leq 10^9$).

The second line of each test case contains n numbers a_i — the elements of the game array ($-10^9 \leq a_i \leq 10^9$).

Output

Print «Yes», if Xenia can complete the level, and «No» otherwise.

Examples

standard input	standard output
4 10 6 7 3 4	Yes
1 7 7	No
6 -3 -3 -9 6 3 -6 0	Yes
6 4 1 5 2 4 33 0	No

Problem F. Fibonacci Chocolates

Time limit: 3 seconds
Memory limit: 512 megabytes

There are n chocolates. The i -th chocolate has size $w_i \times h_i$ and belongs either to Alice or to Bob. Chocolates cannot be rotated by 90° during the game, that is, chocolates of sizes $a \times b$ and $b \times a$ are considered different.

A non-empty subset of these chocolates is chosen; the remaining chocolates are discarded. A game between Alice and Bob is then played using the chosen chocolates.

The players take turns making moves, starting with Alice. On her turn, Alice can do exactly one of the following:

- eat all pieces of one of her chocolates (that is, for some i such that $o_i = 0$, all pieces of this chocolate, whose total area is $w_i \times h_i$ —possibly including pieces obtained earlier by splitting by either player—are removed from the game);
- split one of the current pieces of any chocolate, not necessarily her own. If the piece being split has size $a \times b$, then it can be split into two pieces of sizes $x \times b$ and $(a - x) \times b$, where x is a Fibonacci number such that $0 < x < a$.

On his turn, Bob can do exactly one of the following:

- eat all pieces of one of his chocolates (that is, for some i such that $o_i = 1$, all pieces of this chocolate, whose total area is $w_i \times h_i$ —possibly including pieces obtained earlier by splitting by either player—are removed from the game);
- split one of the current pieces of any chocolate, not necessarily his own. If the piece being split has size $a \times b$, then it can be split into two pieces of sizes $a \times y$ and $a \times (b - y)$, where y is a Fibonacci number such that $0 < y < b$.

The game ends when the current player has no valid moves. The player who cannot make a move loses the game.

Subsets are chosen among the 2^n possible subsets. Two subsets are considered different if there exists an index $i \in \{1, \dots, n\}$ such that the i -th chocolate is present in one subset but absent in the other.

Your task is to compute the number of non-empty subsets of chocolates for which Alice wins, assuming both players play optimally. Output the result modulo 998 244 353.

Input

The first line contains a single integer n ($1 \leq n \leq 100$): the number of chocolates.

The next n lines each contain three integers: w_i, h_i ($1 \leq w_i, h_i \leq 50$), and $o_i \in \{0, 1\}$ — the width and height of the i -th chocolate, and its owner. $o_i = 0$ if the i -th chocolate belongs to Alice, $o_i = 1$ if it belongs to Bob.

Output

Print one integer: the number of non-empty subsets of the chocolates such that Alice wins the game on that subset, modulo 998 244 353.

Examples

standard input	standard output
2 1 1 0 1 1 1	1
4 2 2 0 1 2 1 2 1 1 1 1 0	6

Problem G. Magic Ritual

Time limit: 1 second
Memory limit: 512 megabytes

At the Academy of Archmages, there is a special tradition: every graduate must master the ritual of sorting magical spheres perfectly.

On a long altar, there are n spheres placed from left to right, each numbered. Each sphere has a given strength, and to successfully complete the ritual, they must be arranged in non-decreasing order of strength. To achieve this, spheres can be swapped, and in one action, any two spheres can be exchanged.

However, the magical flow between two positions i and j is unstable, and swapping the spheres at these positions requires spending $(i - j - 2)^2$ units of mana.

The ritual is considered complete when the spheres are arranged in the correct order. You need to conduct the ritual in such a way that the total mana expenditure is minimized.

You will need to answer several independent queries about conducting the rituals.

Input

The first line contains an integer t ($1 \leq t \leq 2 \cdot 10^5$) — the number of rituals.

The descriptions of the rituals follow.

In the first line of each ritual, there is a number n ($1 \leq n \leq 2 \cdot 10^5$) — the number of spheres.

In the second line, n integers a_1, a_2, \dots, a_n are given ($0 \leq a_i \leq 10^9$) — the strength of each sphere.

It is guaranteed that the sum of n across all rituals does not exceed $2 \cdot 10^5$.

Output

For each ritual, output a single integer — the minimum mana expenditure to complete the ritual.

Example

standard input	standard output
3	0
3	1
3 2 1	2
3	
2 1 2	
4	
4 3 2 1	

Problem H. Beaver Dam

Time limit: 2 seconds
Memory limit: 512 megabytes

This is an interactive problem.

The beaver dam is a rectangle consisting of $n \times m$ cells, with secret passages between some neighboring cells that beavers can use to move. The dam has k exits — cells in the rectangle from which one can swim out of the dam.

Scientists are trying to understand the structure of the beaver dam. They have already determined its layout — it is known exactly which neighboring cells have passages. It is also known that the graph of passages forms a tree — meaning there is a unique path through the passages between any two cells, visiting each cell no more than once. Unfortunately, the scientists have not yet been able to learn much about the exits — they only know that $k \geq 1$, meaning there is at least one exit from the dam.

To find an exit from the dam, the scientists decided to use test beavers. Beavers are intelligent animals, and they know the layout of their dam perfectly, including the location of all exits. Since the scientists have only a few beavers left for testing, they can conduct the experiment no more than 48 times.

The experiment works as follows. The scientists take two beavers and release them in the cells (i_1, j_1) and (i_2, j_2) of the rectangular dam. After that, each beaver will swim through the passages to the nearest exit. Everything happens incredibly quickly, so the scientists can only determine which beaver reached the exit faster. If both beavers reach the exit at the same time, the result of the experiment is undefined — meaning either the first beaver or the second could arrive first.

The scientists need to study the beaver dam from the inside, so they need to discover at least one exit from the dam. Help them find it.

Interaction Protocol

At the beginning, your program reads the test parameters.

The first line contains two integers n, m —the dimensions of the dam ($2 \leq n, m \leq 10^6$; $4 \leq n \cdot m \leq 10^6$).

The next n lines contain strings h_i of length $m - 1$ consisting of 0s and 1s, describing the passages in the dam: if $h_{i,j} = 1$, then there is a direct passage between the cells (i, j) and $(i, j + 1)$.

The next $n - 1$ lines contain strings v_i of length m consisting of 0s and 1s, describing the vertical passages in the dam: if $v_{i,j} = 1$, then there is a direct passage between the cells (i, j) and $(i + 1, j)$.

Then your program interacts with the jury's program, making queries and receiving the results of the experiments. You can perform the action described in the statement no more than 48 times.

To ask a question, output a string in the format “? $i_1 j_1 i_2 j_2$ ”, after which 2 beavers will be released into the dam at the specified points (i_1, j_1) and (i_2, j_2) . The conditions $1 \leq i_1, i_2 \leq n$; $1 \leq j_1, j_2 \leq m$ must be satisfied.

In response, the jury's program outputs:

- 1, if the first beaver reaches the exit no later than the second;
- 2, if the second beaver reaches the exit no later than the first;
- Any of the two verdicts above if the beavers reach the exits simultaneously.

To output the answer to the problem, output the string “! $i j$ ”, where i and j should be the row and column numbers where one of the exits is located. This output does not count towards the number of queries. If there are multiple exits in the dam, you can output any of them. After that, the interactor will output a verdict: “Win” if your guess is correct, and “Lose” otherwise. If your answer was incorrect, your solution will receive a verdict of WA (Wrong Answer) and terminate. To avoid receiving incorrect verdicts, your solution should also terminate if it receives information that the outputted answer was incorrect.

If at any point your program exceeds the limit of 48 queries, your program will terminate with a verdict of WA.

After each action of your program, output a newline.

After each action of your program, flush the output stream.

If you are using “`writeln`” in Pascal, “`cout << ... << endl`” in C++, “`System.out.println`” in Java, “`print`” in Python, “`Console.WriteLine`” in C#, then the output stream is flushed automatically, and no additional action is required. If you are using another method of output, it is recommended to flush the output stream.

Note that in this problem, the interactor may be adaptive, meaning the state of the maze always corresponds to the queries already made, but otherwise may change during the execution.

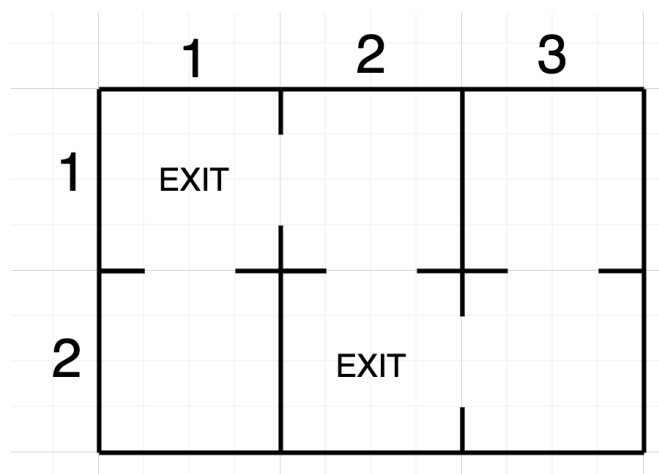
Example

In sample test interaction messages between the jury program and the contestant program are separated by empty lines to visualize which message is a response to which. In real interaction there will be no empty lines and you should not print any.

standard input	standard output
2 3	
10	
01	
111	
	? 1 1 1 2
1	
	? 1 3 2 1
2	
	? 1 2 2 3
1	
	? 1 2 2 3
2	
	! 2 2
Win	

Note

In the example from the statement, the maze looks like this:



That is, the exits in the dam are located at points (1,1) and (2,2).

Problem I. Predicting a Position

Time limit: 1 second
Memory limit: 512 megabytes

An ascending sorted array of unique integer keys x_i and some integer non-negative constant ε is given.

It is necessary to split the original key array into a minimum number of sub-arrays, so that on each sub-array $[l \dots r]$ there is some linear function $f(x) = k \cdot x + b$, predicting for the key x_i its position on this sub-array — it is equal to $i - l$ — with an error not exceeding ε .

Formally, for the j -th of the sub-array, the coefficients k_j and b_j must exist (not necessarily integers) so that for all the keys x_i on that sub-array $i \in [l_j \dots r_j]$ is true:

$$|f_j(x_i) - (i - l_j)| \leq \varepsilon$$
$$f_j(x) = k_j \cdot x + b_j$$

Input

In the first line of the input data, two integers are separated by a space: n, ε — the number of keys and the maximum allowable error, respectively ($2 \leq n \leq 10^6, 0 \leq \varepsilon \leq n$).

The second line of input data contains the unique keys x_i separated by a space in ascending order ($-2 \cdot 10^9 \leq x_1 < x_2 < \dots < x_n \leq 2 \cdot 10^9$).

Output

In a single line of the output data, print a single integer m — the minimum number of sub-arrays into which you were able to split the original array of keys so as to meet the requirements.

Example

standard input	standard output
8 0 1 2 3 4 7 10 13 16	2

Note

In the example, you can split the specified array of keys into two: $[1, 2, 3, 4]$ and $[7, 10, 13, 16]$. In the first case, the key position is predicted exactly by the function $f(x) = x - 1$. In the second case, the key position is accurately predicted by the function $f(x) = (x - 7)/3 = 1/3 \cdot x - 7/3$.

Problem J. Strange Sum

Time limit: 1 second
Memory limit: 512 megabytes

Given two non-negative integers n and x .
Also, for all $1 \leq l \leq r \leq n$, an integer $w_{l,r}$ is given.
For an array of integers $A = [a_1, a_2, \dots, a_n]$, define

$$f(A) = \sum_{l=1}^n \sum_{r=l}^n w_{l,r} \cdot \min(a_l, a_{l+1}, \dots, a_r).$$

You are required to find the maximum possible value of $f(A)$ over all arrays A such that

$$a_1 + a_2 + \dots + a_n = x \quad \text{and} \quad a_i \geq 0.$$

Input

The input in this problem contains one or more test cases.

The first line contains a single integer t – the number of test cases ($1 \leq t \leq 50$).

The descriptions of the t test cases follow.

The first line of each test case contains two integers n and x ($1 \leq n \leq 50$, $1 \leq x \leq 10^9$).

In the i -th of the following n lines, there are $n - i + 1$ integers $w_{i,i}, \dots, w_{i,n}$ ($1 \leq w_{i,j} \leq 10^6$).

Output

For each test case, output a single integer: the maximum value of $f(A)$ among all arrays A such that

$$a_1 + a_2 + \dots + a_n = x \quad \text{and} \quad a_i \geq 0.$$

Example

standard input	standard output
3	30
5 10	10
1 1 1 1 1	14999999995
1 1 1 1	
1 1 1	
1 1	
1	
1 10	
1	
4 1000000000	
1 2 3 4	
5 6 7	
8 9	
10	

Problem K. Two Skew Roads

Time limit: 1 second
 Memory limit: 512 megabytes

While reading the statements of yet another boring problem in some contest (unrated, teams preferred) that some people called incredibly important, you fell into nostalgia and thoughts about your hometown, Berlodar. You remembered how you enjoyed walking along its two main roads — Holmes Avenue and Watson Street, which divide the city into four huge districts. You smirked slightly when you recalled how after some long-ago lesson in fifth grade, when your math teacher told you about angles and distances, your eyes lit up, and for the entire evening that followed, you walked back and forth around the city's main intersection and found out that the width of Holmes Avenue is ℓ_H meters, the width of Watson Street is ℓ_W meters, and they intersect at an angle of α degrees. You dug up a map of the city at home and were pleased to confirm that your measurements were correct.

Soon you will return to Berlodar for the New Year holidays and want to visit your four friends. They all live in different districts, so you will have to cross a main road many times, and sometimes both, to visit all your friends. While planning your walk, you will wonder: what distances will you have to cover between these four districts? The traffic light regime on these two streets implies that the green light for all pedestrians (and, consequently, the red light for vehicles on both roads) turns on simultaneously, so to get between any pair of districts (even those located opposite each other), you will walk along the shortest segment connecting their boundaries.

There are $\frac{4 \cdot (4-1)}{2} = 6$ pairs of districts. Write a program that prints six numbers equal to the pairwise distances between the districts in Berlodar.

Input

The input consists of a single line containing three real numbers ℓ_H , ℓ_W , and α , denoting the width of Holmes Avenue in meters, the width of Watson Street in meters, and the angle at which they intersect in degrees ($0.01 \leq \ell_H, \ell_W \leq 1000$; $0.1 \leq \alpha \leq 90$). Each of these three numbers will be provided with a precision of up to three decimal places.

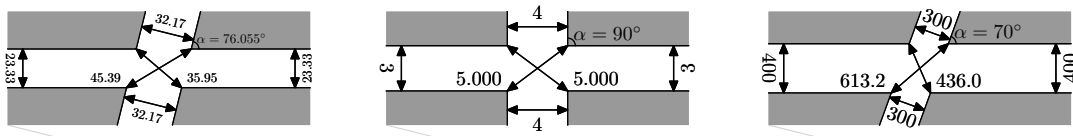
Output

Print six real numbers in any order — the pairwise distances between the four districts of Berlodar in meters. Your answer will be accepted if the relative or absolute error does not exceed 10^{-6} . Formally, if $\{a_i\}_{i \in [1..6]}$ is your answer and $\{b_i\}_{i \in [1..6]}$ is the jury's answer, it will be accepted if the numbers in array a can be rearranged such that after rearrangement, the condition $\frac{|a_i - b_i|}{\max\{b_i, 1\}} \leq 10^{-6}$ holds for each index $i \in [1..6]$.

Examples

standard input	standard output
23.332 32.17 76.055	23.332 23.332000 32.17 32.170 35.95260 45.39539
3 4 90	5 4 3 3 4 5
400 300 70	400 300 400 300.0 613.251621126 436.073

Note



Problem L. System of Equations with XOR

Time limit: 1 second
Memory limit: 512 megabytes

Alice and Bob love problems involving random numbers. One day they came up with the following problem:

- First, Alice chooses a random integer x from the range of 1 to $2^{31} - 1$, all numbers are equally likely.
- Then, Bob chooses a random integer y from the range of 1 to $2^{31} - 1$, all numbers are equally likely.
- They calculate the product of these numbers $a = x \cdot y$ and their bitwise XOR $b = x \oplus y$.

You are given two resulting integers a and b . Find any pair of natural numbers x and y such that:

$$xy = a \quad \text{and} \quad x \oplus y = b,$$

where \oplus denotes the bitwise XOR operation.

Recall that the bitwise “exclusive or” (\oplus , xor) of two non-negative integers is defined as follows: write both numbers in binary representation; the i -th binary digit of the result is 1 if exactly one of the arguments has it equal to 1. For example, $(14 \text{ xor } 7) = (1110_2 \oplus 0111_2) = 1001_2 = 9$. This operation is implemented in all modern programming languages; in C++, Java, and Python, it is written as “`^`”, and in Pascal as “`xor`”.

Input

The first line of the input contains a single integer t ($1 \leq t \leq 200\,000$) — the number of test cases.

In the following t lines of input, there are two integers a and b ($1 \leq a < 2^{62}$, $0 \leq b < 2^{31}$) — the description of the next test case.

Output

For each test case, output in a separate line two natural numbers x and y , separated by a space, such that $xy = a$ and $x \oplus y = b$.

If there are multiple valid answers, you may output any of them.

Example

standard input	standard output
2	7 3
21 4	3 3
9 0	

Note

In this problem, there are 100 tests, including the example from the statement. It is guaranteed that in all tests, except for the example from the statement, $t = 200\,000$, and the numbers a and b for each test case were generated by choosing random numbers x and y by Alice and Bob.

Problem M. Mathematical Calendar

Time limit: 1 second
Memory limit: 512 megabytes

Lusya attends a programming club on Thursdays and a mathematics club that meets on Mondays. For New Year, students who excel in their clubs were given a beautiful pocket calendar as a gift. Lusya received one as well.

Lusya loves to look for different patterns. After studying the calendar, she discovered that in the new year, the number of winter Thursdays is 2 more than the number of winter Mondays. Lusya realized that if she knows this condition about the year, as well as the fact that the new year will not be a leap year, she can uniquely determine the day of the week for any given day and month. Lusya solved this problem and offers you to solve it as well.

Let us remind you that the winter months are December, January, and February. January, March, May, July, August, October, and December have 31 days, April, June, September, and November have 30 days, and February has 28 days in a non-leap year.

Input

The input consists of a single line containing a string — a date in the format DD-MM.

Output

Output an integer from 1 to 7 — the day of the week that corresponds to the given date, 1 being Monday and 7 being Sunday.

Example

standard input	standard output
16-12	3