The International Collegiate Programming Contest
Sponsored by ICPC Foundation

The 2020 Africa and Arab Collegiate Programming Championship

(Contest Problems)

Arab Academy for Science and Technology
and Maritime Transportation
Luxor
March 2021

Problem A. Arc Measure

Input file:	`arc.in`
Output file:	`standard output`
Balloon Color:	`White`

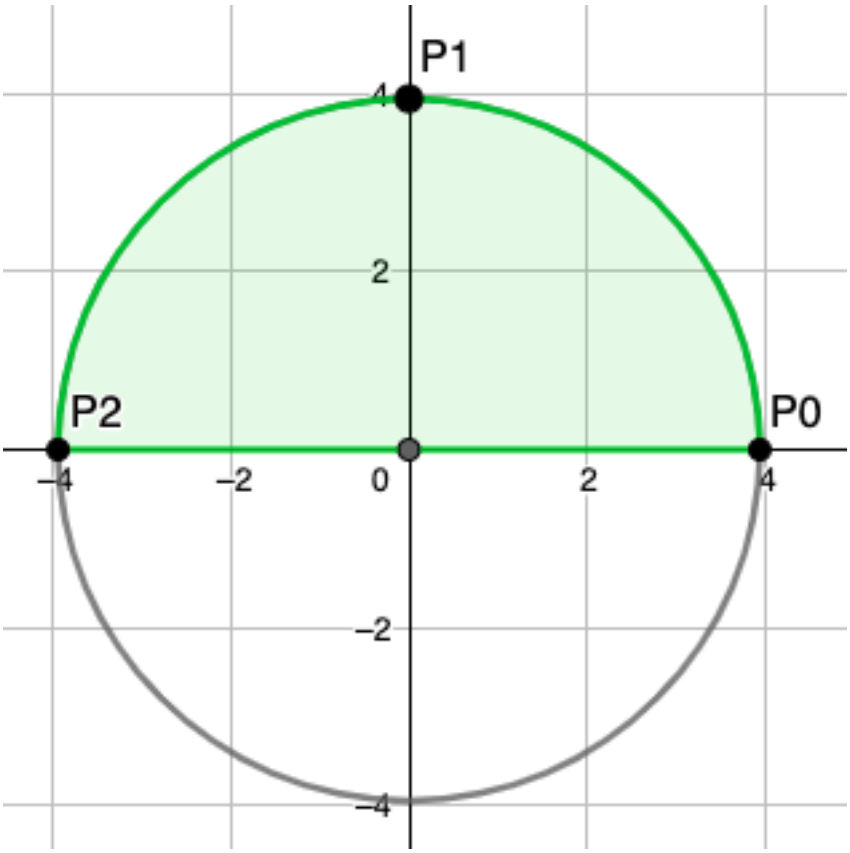
Coach Academy teaches students to think, create and code. One day the instructor gave the students this interesting problem.

If you were one of these students, would you be able to solve it?

Given a set of N points on the circumference of a $2D$ circle centered at the origin, where the i^{th} point is located at a_i degrees from the $+ve$ x-axis.

You need to find the minimum integer arc measure M, such that you can covered all the points using no more than K arcs of measure M. The arc covering must be contiguous with no gaps.

It is ok for two or more arcs to overlap.



Input

First line will be the number of test cases T, in each test case you will be given the following:

First line will have 2 integers N and K separated by a space. $(1 \leq N, K \leq 360)$

Followed by a line containing N integers separated by a space, each integer $(0 \leq A_i \leq 359)$

Output

For each test case, output one number in a line by itself, M the minimum arc measure.

Example

arc.in	standard output
1 3 1 0 90 180	180

Note

An arc measure is the number of degrees it can cover from the circumference. For example, an arc measure of 90 covers quarter of the circle.

Problem B. Basketball

Input file:	`basket.in`
Output file:	`standard output`
Balloon Color:	`Purple`

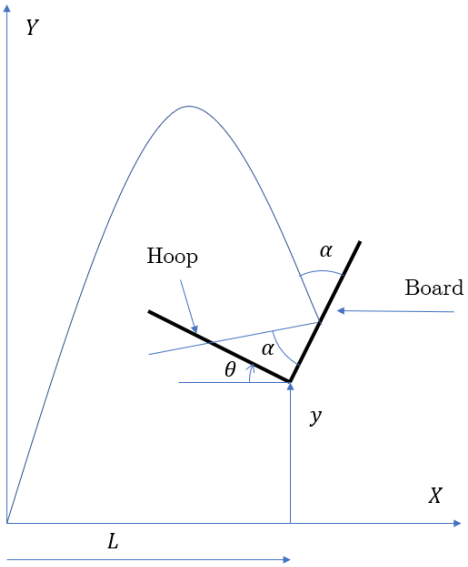
You are developing a software for a smart basketball backboard.

A player will be shooting a ball (for simplicity, we will consider it to be a size of a point) from the origin of a 2D frame. A square board of side D is located at $x = L$ and a circular hoop of diameter D is attached to that board. In the XY plane, the board and the hoop look like two perpendicular line segments of length D each, parallel to the Y and X axes, respectively. They both can move along the Y^+ axis and rotate around the Z axis without exceeding ± 60 degrees (positive rotation is in clockwise direction).

To score a point the ball has to go through the hoop after an impact with the midpoint of the board. If the ball hits the board with an angle α, it will bounce and follow a linear trajectory that makes an angle of $\pi - \alpha$ with the board.

You know that the ball will have a parabolic trajectory ($Ax^2 + Bx$).

Output a state of the board (position and angle) allowing the player to score a point or say if it is impossible.



Input

First line will be the number of test cases T, in each test case you will be given the following:

One line containing four integers: L, D, A, B separated by spaces. $-10^9 \leq A \leq 0, -10^9 \leq B \leq 10^9$, $0 \leq L \leq 10^9$ and $0 < D < L$

Output

For each test case, on a new line, if it is possible to score a point, output 2 numbers : y the position and θ the angle of the board (refer to the figure, output angles in radians). Otherwise output -1.

The answer (y, θ) is considered correct if after hitting the board with an angle α, at a point p distant from the midpoint by at most 10^{-4}, the ball will go through the hoop without intersecting any endpoint of the line segment representing the hoop, following a line that makes an angle $\beta \in [\pi - \alpha - 10^{-4}, \pi - \alpha + 10^{-4}]$ with the board.

Example

basket.in	standard output
2 2 1 -1 5 14 5 -1 25	-1 154.3340548626 -0.7071634234

Note

It is possible for the ball to go through the hoop from below, hit the board and bounce back to score a point.

Problem C. Completions

Input file:	`completions.in`
Output file:	`standard output`
Balloon Color:	`Gray`

You are given 2 strings s_1 and s_2.

For each unique substring present in both strings, find the number of unique completions for it present in both string. Then, output the sum of squares of each substring's answer.

A completion for a substring *sub* in a string S is a non-empty substring *compl* such that the string $sub + compl$ is a substring of S.

Let us take this example:

s_1 = "ABCDABQW"

s_2 = "ABCGABQW"

The completions for substring "A" in s_1 are {"B", "BC", "BCD", "BCDA", "BCDAB", "BCDABQ", "BCDABQW", "BQ", "BQW"}.

While the completions for the same substring "A" in s_2 are {"B", "BC", "BCG", "BCGA", "BCGAB", "BCGABQ", "BCGABQW", "BQ", "BQW"}.

The common completions for the substring "A" in both strings are {"B", "BC", "BQ", "BQW"}. So, there are 4 completions for the substring "A" that are common in both strings.

You need to count the completions for every unique substring, and output the sum of squares modulo 132120577.

Input

First line will be the number of test cases T, in each test case you will be given the following:

The first line will have two integers separated by a space N_1 and N_2, the lengths of the strings s_1 and s_2. where $(1 \leq N_1, N_2 \leq 10^5)$.

The second line will have the string s_1 of length N_1.

The third line will have the string s_2 of length N_2.

Strings will contain only uppercase English letters.

Output

For each test case, output 1 integer in a line by itself, the answer to the problem.

Example

completions.in	standard output
1 8 8 ABCDABQW ABCGABQW	37

Problem D. Differences

Input file:	`differences.in`
Output file:	`standard output`
Balloon Color:	`Dark green`

You have an array of pairs of integers (X_i, F_i) where F_i is the frequency of X_i in some very big array A.

You will be given Q queries to be performed on A, each query will have 2 integers l and r.

For each query, output the number of pairs of integers (A_i, A_j) with an absolute difference between l and r (inclusive). In other words, the number of pairs (A_i, A_j) such that $l \leq |A_i - A_j| \leq r$.

Since this number is large, output it modulo 132120577.

Input

First line will be the number of test cases T, in each test case you will be given the following:

First line containing an integer N, the number of frequency pairs. $(1 \leq N \leq 10^5)$

Followed by N lines where each line has 2 integers separated by a space: X_i and F_i. $(1 \leq X_i \leq N)$ and $(1 \leq F_i \leq 10^9)$

Followed by a line containing an integer Q, the number of queries. $(1 \leq Q \leq 10^6)$

Followed by Q lines, each line containing 2 integers l and r separated by a space. $(1 \leq l, r \leq N)$

Output

For each test case, output Q integers, each integer on a separate line.

Each line should contain 1 integer, the answer to the i^{th} query.

Example

differences.in	standard output
1 3 1 1 2 1 3 1 1 1 3	3

Problem E. Endgame

Input file:	`endgame.in`
Output file:	`standard output`
Balloon Color:	`Orange`

A rook endgame is a position in chess where all the pieces were captured, and the only remaining pieces are the 2 kings and 1 rook (in our case, white is the one having the rook).

Although this end game is winning for white, it is not that easy to actually win this endgame. White needs to know exactly what they need to do. Otherwise, it is easy for black to draw this endgame.

As many may know, the king in chess can move only one square at a time in any of the 8 directions, while the rook can move any number of squares, but only vertically and horizontally.

Chess board consist of 8×8 squares, the rows are numbered 1 to 8, and the columns are numbered 'a' to 'h'.

The position of a square is denoted by the letter of the column followed by the number of the row. For example "c5" is the square in the 3^{rd} column and the 5^{th} row.

Given the position of the white king, the white rook, and the black king. It is black to move. Your task is to output whether black's king is under attack by white's rook or not.

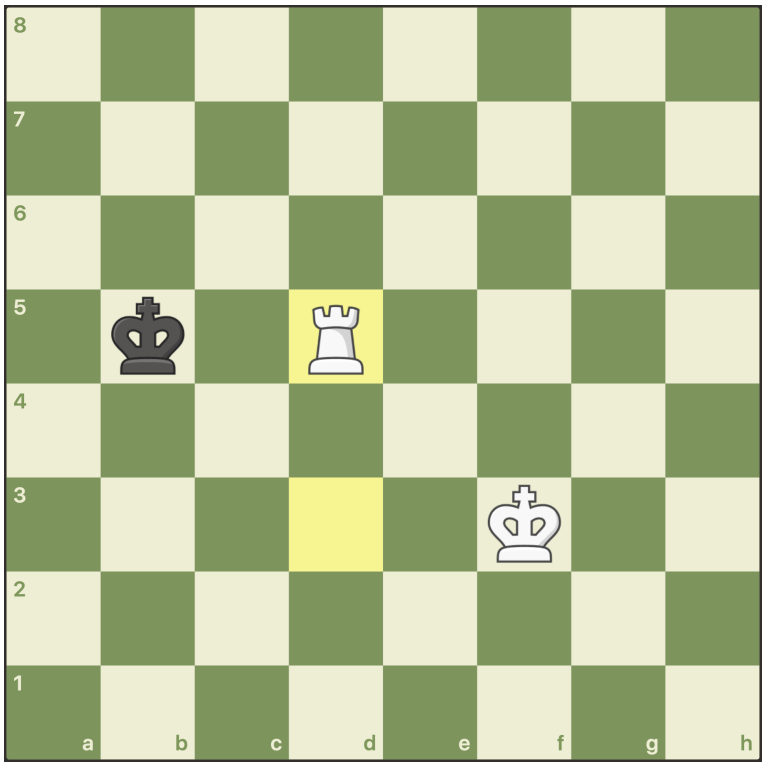


Рис. 1: In this figure the black king is attacked.

Input

First line will be the number of test cases T, in each test case you will be given the following:

One line containing 3 strings separated by spaces. The position of the white king, white rook, and black king in that order.

Output

For each test case output a single line containing "YES" if the black king is attacked by the rook, or "NO" otherwise.

Example

endgame.in	standard output
1 f3 d5 b5	YES

Problem F. Find the variance

Input file:	`variance.in`
Output file:	`standard output`
Balloon Color:	`Blue`

Consider a binary string s.

Let Y be the number of alternations in s, i.e. the number of bits following a bit of different value.

For example: if $s = 001100010100111$, then $Y = 7$.

Each bit in s, s_i is be 1 with probability p_i and zero with probability $1 - p_i$

Given the probability array p. Find the variance of Y.

Input

First line will be the number of test cases T, in each test case you will be given the following:

First line will contain an integer N, the length of the string. ($1 \leq N \leq 10^5$)

Second line will contain N positive floating point numbers separated by spaces, the i^{th} number is the probability of bit i being a 1. All numbers on this line are ≤ 1.

Output

For each test case, output one number, the variance of Y.

Example

variance.in	standard output
4 1 0.5 2 0.5 0.5 2 0.1 0.3 5 0.1 0.2 0.3 0.4 0.5	0.000000 0.250000 0.224400 1.079200

Note

The variance can be computed with the formula: $Variance(Y) = Expectation(Y^2) - Expectation(Y)^2$

Problem G. Graph colourings

Input file:	`onion.in`
Output file:	`standard output`
Balloon Color:	`Red`

An onion graph is a collection of cycles $C_1, C_2, ..., C_k, ..., C_2, C_1$ with some extra edges. We call each cycle a 'layer'.

Each layer is a cycle with the subscript number of nodes. For example, for k = 3, the graph is composed of C_1, C_2, C_3, C_2, C_1 (one node, two connected nodes, a triangle of three connected nodes, two connected nodes and finally, one node).

In addition to the edges in each cycle, a node in any layer can be connected by a single edge to nodes of adjacent layers. Also, the nodes in the C_1 layers can be connected by a single edge. However, nodes from non-adjacent layers cannot be connected by a single edge.

Given an onion graph with $1 \leq k \leq 5$, count the number of 4-colorings. Since this number can be large, print the result mod 132120577.

A 4-coloring is an assignment of colors (red, green, yellow and blue) to nodes, such that if two nodes are connected by an edge, they must have different colors.

Input

First line will be the number of test cases T, in each test case you will be given the following:

The first line contains three integers separated by spaces n, k, m. ($1 \leq k \leq 5$)

Where n is the total number of nodes, k is such that the widest middle layer has k nodes and m is the number of edges, the number of edges will be at least the edges required to make the cycles of a correct onion graph, but nodes of adjacent cycles can be fully connected.

Followed by m lines, each contains a pair of integers separated by spaces, denoting 2 nodes to be connected by a single edge.

The nodes are numbered layer by layer starting at 1 (first C_1 has node 1, second layer C_2 has nodes 2 and 3, ..etc.).

It is guaranteed that the input is a valid onion graph and that $n = k^2$.

Output

For each test case, output the number of 4-colorings mod 132120577 in a line by itself.

Example

onion.in	standard output
1 9 3 5 2 3 4 5 5 6 6 4 7 8	55296

Problem H. Hash collision

Input file:	`hash.in`
Output file:	`standard output`
Balloon Color:	`Cyan`

Consider the following polynomial function $f(S)$ used to hash a string S_i of length k.

$f_i(S_i) = a_{i,k} * S_{i,k}^k + a_{i,k-1} * S_{i,k-1}^{k-1} + a_{i,k-2} * S_{i,k-2}^{k-2} + ... + a_{i,2} * S_{i,2}^2 + a_{i,1} * S_{i,1} \pmod m$

In other words, $f_i(S_i) = \sum_{j=1}^{k} a_{i,j} * S_{i,j}^j \pmod m$

Where $a_{i,j}$ are integer coefficients, m is the modulo, and $S_{i,j}$ represents the alphabet order of the j^{th} character in S_i.

For example, if S_i = "abz", then $S_{i,1} = 1$, $S_{i,2} = 2$, $S_{i,3} = 26$ (i.e. 'a' = 1, 'b' = 2, .., 'z' = 26).

Given a list of n strings of equal length k, consisting of lowercase letters, your task is to find any list of integer coefficients $a_{i,j}$ $(1 \le i \le n, 1 \le j \le k)$ that guarantees no collisions occur.

In other words, you need to find any list of coefficients such that there does not exist a pair of strings S_i and S_j where $i \neq j$ and $f_i(S_i) = f_j(S_j)$.

Notice that you do not have to use the same hashing function for all the strings. Instead, you can choose a completely different function (list of coefficients) to hash each string.

Input

First line will be the number of test cases T, in each test case you will be given the following:

First line will be 3 integers n, k, m separated by spaces. $(2 \le n, k, m \le 500)$

Then n lines follow, each line has a string of length k consisting of lowercase english letters.

Output

For each test case:

If it is impossible to avoid collisions, output "No" in a single line.

Otherwise output "Yes" followed by n lines.

On the ith line, output k integers $a_{i,1}$ $a_{i,2}$ .... $a_{i,k}$ separated by spaces.

Example

hash.in	standard output
1 5 5 11 abcde fghij klmno pqrst uvwxy	Yes 4 0 0 0 0 3 8 0 0 0 0 2 0 0 0 10 2 0 0 0 0 0 0 0 0

Problem I. Inside the circle

Input file:	`circles.in`
Output file:	`standard output`
Balloon Color:	`Gold`

You are given a circle C, a set of "obstacle" circles $O_1, ..., O_n$ and two points S and E.

Count the minimum number of obstacles you will have to go through by taking a valid path from S to E.

A path is valid if all the points on the path lie inside C.

The following properties are guaranteed:

- The points S and E are inside C.
- All obstacles are either inside C or intersect with C.
- Each obstacle will be intersecting with at most two other circles (can be C or any of the obstacles).
- No obstacle will contain another obstacle.

Input

First line will be the number of test cases T, in each test case you will be given the following:

First line contains four integers describing the coordinates of the points S and E. ($-10^9 \leq S_x, S_y, E_x, E_y \leq 10^9$)

Second line contains an integer n, the total number of circles (the big circle + (n-1) obstacles). ($1 \leq n \leq 10^4$)

Followed by n lines, each describing a circle (first C, followed by the n-1 obstacles).

Each line contains three integers r, x, and y, describing the radius of the circle and coordinates of the center.

All integers on the same line are separated by spaces.

Output

For each test case output one line containing the number of obstacles you have to go through to pass from S to E.

Example

circles.in	standard output
1 0 2 0 -2 4 3 0 0 1 0 0 1 -2 0 1 2 0	1

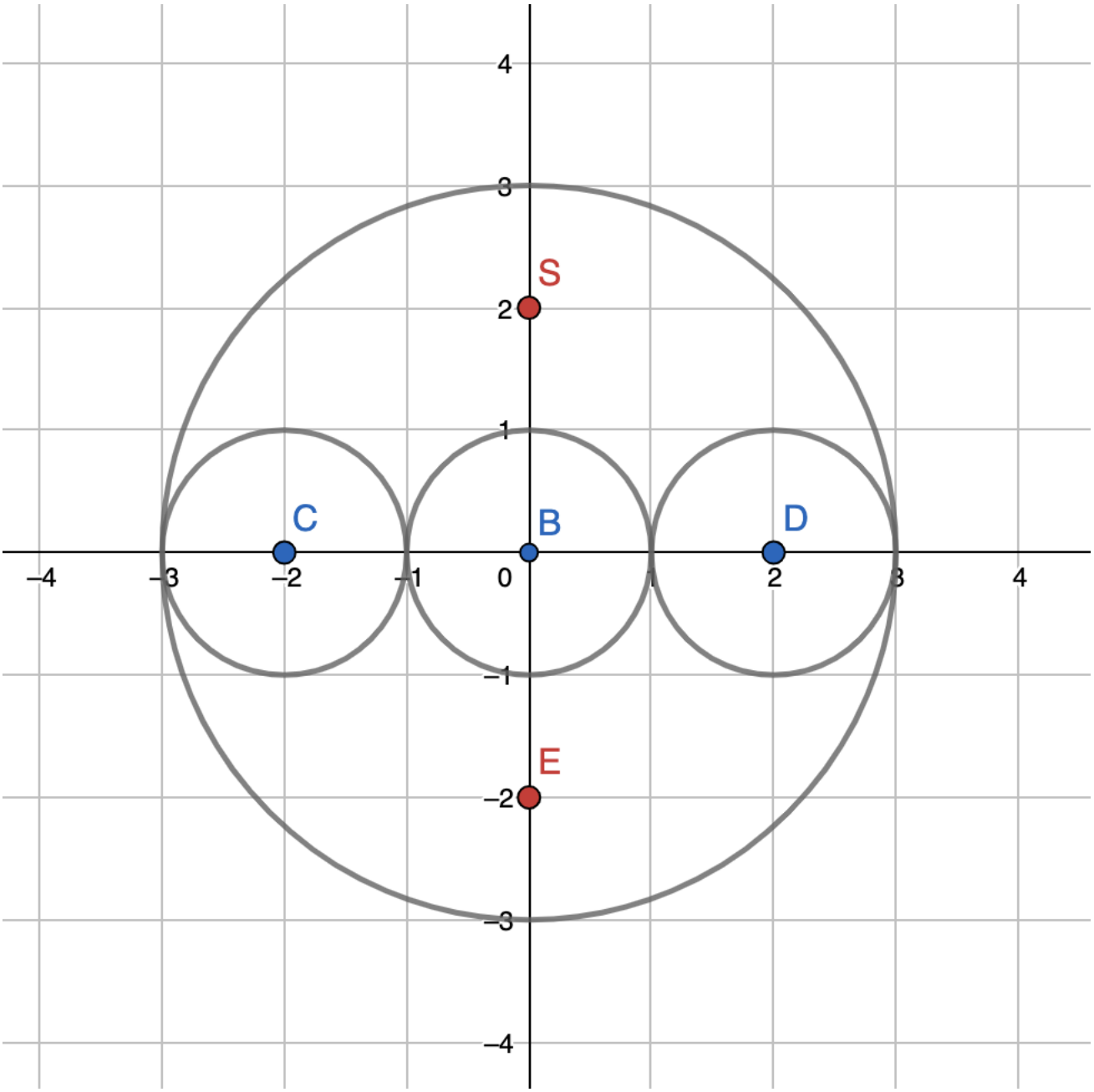


Рис. 2: The valid path from S to E can pass through at least one obstacle.

Problem J. Judging sums

Input file:	`judging.in`
Output file:	`standard output`
Balloon Color:	`Yellow`

There are M persons, each with a list of positive integers.

They form 2 arrays, A and B by following these steps:

1. Add their numbers into A.
2. Sum their numbers and add the sum to B.

For example, if we have 3 persons, p_1, p_2 and p_3:

- p_1 has [2, 1].
- p_2 has [2].
- p_3 has [2].

The final arrays will be:

- A = [2, 1, 2, 2].
- B = [3, 2, 2].

Then they shuffle both A and B

You are given the 2 arrays A and B, for each pair of elements x, y where $x \in A$ and $y \in B$, find the probability that they were added by the same person.

Input

First line will be the number of test cases T, in each test case you will be given the following:

The first line contains 2 integers separated by a space N and M. ($1 \leq M \leq N \leq 15$)

Followed by a line containing a list of N positive integers describing A. ($1 \leq A_i \leq 2 \times 10^9$)

Followed by a line containing a list of M positive integers describing B.

Integers that are in the same line are separated by spaces.

Output

For each test case, output $N * M$ matrix (numbers in the row separated by spaces and each row in a line by itself).

Each entry in the matrix is the answer for the pair (A_i, B_j) for every $1 \leq i \leq N$ and $1 \leq j \leq M$.

Example

judging.in	standard output
1 4 3 2 1 2 2 3 2 2	0.3333 0.3333 0.3333 1.0000 0.0000 0.0000 0.3333 0.3333 0.3333 0.3333 0.3333 0.3333

Problem K. Kill the dragon, or run

Input file:	`dragon.in`
Output file:	`standard output`
Balloon Color:	`Light green`

You are running from a dragon:

- You have speed S_1.
- The dragon has speed S_2.
- You start at position P.
- The dragon starts at position 0 and starts running towards you immediately.

You have a sword. When the dragon reaches you, you can hit it with your sword, which injures it.

After getting hit, the dragon stops running and stays in its position for H seconds to heal, then continues running after you.

Your sword can only hit the dragon X times before it breaks. If the dragon reaches you while your sword is broken, it will eat you.

You survive if you reach position Z before the dragon eats you. Calculate the minimum number of sword hits you need to survive or state that it is impossible.

Input

First line will be the number of test cases T, in each test case you will be given the following:

Five integers S_1, S_2, P, X, H, Z separated by spaces such that $(1 \leq S_1, S_2, P, H \leq 10^9)$, $(0 \leq X \leq 10^9)$ and $(P \leq Z \leq 10^{18})$

Output

For each test case, if you can survive output the minimum number of sword hits you need to do in order to survive, otherwise print "Impossible".

Output one line for each test case.

Example

dragon.in	standard output
3 1 2 1 1 2 10 3 2 2 1 2 20 1 39 20 9 4 28	Impossible 0 2

Problem L. Looking for pattern

Input file:	`pattern.in`
Output file:	`standard output`
Balloon Color:	`Pink`

Let us have yet one more string matching problem.

Two strings s_1 and s_2 are considered equal if and only if they have the same probability to occur as a substring in a random string R of a significantly bigger size (at least $2 \times max(|s_1|, |s_2|)$).

Similarly $s_1 < s_2$ if and only if s_1 is less probable than s_2 to occur as a substring in the large random string.

We want to solve the classical pattern matching problem.

Given a text T, and a pattern P, we want to know how many times does P occur in T and the indices of the beginning of these occurrences.

However, let us define how the text and pattern are described:

- The text T is an array of strings. (i.e. [“abcd”, “aaaa”, “abab”, “aaba”])
- The pattern P is an array of strings. (i.e. [“bbbb”, “bbab”])

Two arrays are equal if the strings inside them have the same order according to the comparison definition above.

In the pattern described above, “bbbb” is smaller than “bbab”, and P occurs twice in T, the first occurrence at position 1 and second occurrence at position 2 (0-based).

Input

The first line will be the number of test cases T, in each test case you will be given the following:

The first line will have 2 integers, T and P separated by spaces (the lengths of the text and the pattern).

The second line will have T space-separated strings, denoting the array that represents the text.

The third line will have P space-separated strings, denoting the array that represents the pattern.

It is guaranteed that the total number of characters given will not exceed 10^7.

It is guaranteed that all the strings have the same length and all characters are lower-case English letters.

Output

For each test case output 2 lines:

On the first line, the number of occurrences of the pattern in the text.

On the second line, the indices of the start of the pattern’s occurrences in the text separated by spaces (0-based).

Example

pattern.in	standard output
1 4 2 abcd aaaa abab aaba bbbb bbab	2 1 2

Note

This note is to clarify the matching: if you have 4 strings in this order [second smallest, smallest, smallest, largest] then you need to match any sub-array that matches that pattern, regardless of what the strings actually are.

The comparison is done by the definition above.

Problem M. Making money

Input file:	money.in
Output file:	standard output
Balloon Color:	Black

Endure Capital is an early stage investment fund headed by entrepreneurs. Unlike many venture capital firms, Endure prides itself on being founder-focused. It believes in the people behind the idea, and works with them through challenges, hurdles and objectives.

Endure is making a research to find out which countries contain startups that make more money, so that they invest in these countries.

You are given a list of country names, along with the profit each country makes in millions of dollars, output the name of the country that makes most money.

Input

First line will be the number of test cases T, in each test case you will be given the following:

First line will be N, the number of countries. ($1 \leq N \leq 100$)

Followed by N lines, each line has a string C (a country name) followed by an integer P (the profit of this country in millions of dollars), separated by a space. ($1 \leq P \leq 1000$)

The length of C will not exceed 10 characters.

It is guaranteed that all countries are unique, and that each country has a unique profit.

Output

For each test case, output the name of the country that makes most money.

Example

money.in	standard output
1 3 Sudan 300 Egypt 500 Tunis 400	Egypt