Problem A. Anime

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Aizhan is watching the last episode of the JJK. The duration of the last episode is exactly n seconds. For every moment, she knows the level of interest: $I(t)$. Fortunately, function $I(t)$ is a continuous piecewise linear function. She knows the values of $I(t)$ at integer moments of time: $I(0), I(1), I(2), I(3), \ldots, I(n)$. To get values at non-integer moments, she can draw the graph of the function I with points $(x, I(x))$ for every integer x from 0 to n, and then connect consecutive points by straight line segments. The interest of watching a fragment of the episode is the area under the graph for that fragment.

The video player is very strange. It has only two buttons: fast forward and rewind. The first button will forward the video by k seconds, and the second button will rewind the video by k seconds. So it is not possible to stop the video.

By trial and error, Aizhan noticed that it is not possible to forward the video or rewind it if she ends up outside of the episode's domain; in other words, she must stay in the time interval $[0, n]$ (domain expansion is not possible). The second peculiarity is the fact that the number of uses for both buttons should be equal after she has finished the episode; otherwise, her computer will blow up, which is not fun at all.

Aizhan has finished watching the episode without using the buttons, and the cumulative interest she got is the area under the graph of function I. Now, she wonders what is the maximum possible cumulative interest if she uses the buttons optimally.

Note that the buttons can be used at any moment, even at non-integer moments. The buttons can be used any number of times, as long as, at the end of the episode, the first button is used the same number of times as the second one.

Input

On the first line, you are given two integers n and k ($1 \le k \le n \le 10^5$): the duration of the episode and the characteristic of the video player.

On the second line, you are given $n + 1$ integers $I(0), I(1), I(2), I(3), \ldots, I(n)$ ($0 \le I(i) \le 10^5$): the level of interest at all the integer moments.

Output

In the single line, output one number: the answer for the problem.

Your answer is considered correct if its absolute or relative error does not exceed 10^{-6}.

Examples

standard input	*standard output*
2 1 0 1 2	3.000000
2 1 0 5 0	7.500000

Note

In the first example, Aizhan can use the forward button at moment 0, watch the time interval $[1, 2)$ of the video, then rewind it to watch the time interval $[1, 2)$ again.

Note that Aizhan watched time interval $[1, 2)$ twice, and the area under this interval on the graph is 1.5. Thus, the cumulative interest is 3.0.

Problem B. Random Interactive MST Bot

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

How do setters come up with problems? Sometimes they just take a couple of buzzwords and smash them together. But we are in 2024, so this totally can be outsourced to AI. Introducing our creation based on ChatGPT: RICH B! And its second official problem:

Prompt: Minimum Spanning Tree

Problem: A complete graph with n nodes and $\frac{n \cdot (n-1)}{2}$ edges is chosen. Each edge is randomly assigned a real-valued weight within the range of $[0, 1]$. Your task is to find its minimum spanning tree. But you are not given the edges. Instead, you can make queries of the form "`?` v_1 u_1 v_2 u_2", and the jury program will respond to you with 1 if the weight of edge (v_1, u_1) is less than the weight of edge (v_2, u_2), and it will respond with 0 if it's not.

When you think that you know the minimum spanning tree, print it as "`!` v_1 u_1 v_2 u_2 ... v_{n-1} u_{n-1}", where edges (v_i, u_i) form the minimum spanning tree. Constraints: $2 \le n \le 100$, and you can make at most 6000 queries.

Interaction Protocol

First, read a line that contains a single n ($2 \le n \le 100$): the size of the graph.

To compare two edges, print a single line in the following format: "`?` v_1 u_1 v_2 u_2". You will then have to read a line with the result of the comparison: 1 if the weight of the first edge is less than the weight of the second edge, or 0 otherwise.

To output the answer, print a single line in the following format: "`!` v_1 u_1 v_2 u_2 ... v_{n-1} u_{n-1}". Your program has to immediately terminate after printing this line, otherwise, you may get unpredictable verdicts.

In every line you print, (v_i, u_i) should be pairs of integers where $1 \le v < u \le n$), otherwise, you may get unpredictable verdicts.

Your program should not make more than 6000 comparisons.

Example

standard input	*standard output*
3 1 1	? 1 2 1 3 ? 1 3 2 3 ! 1 2 1 3

Note

The Minimum Spanning Tree (MST) of a graph is defined as the graph's spanning tree having the minimum possible total weight.

A spanning tree is a connected subgraph of the given graph that contains all of the graph's vertices and does not contain cycles.

The interactor is **not adaptive**.

Remember to end the line and flush the output after every line you print. To flush the output, you can use `fflush(stdout)` in C/C++, `System.out.flush()` in Java, or `sys.stdout.flush()` in Python.

Problem C. Dota 322 for Droids

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

DotA is a sick mental game. It brings out the best and worst in you. It comes down to half a second. You can't just predict what's going to happen. You try to look at everything as pieces and pawns. You try to look at it as simply "The Game". Who ended up winning the mental warfare always win the series. That's how you're going to win tournaments. If you break the other mind it gets really easy... Some minds are really hard to beat though.

Droid N4-2A was lost for days because he started playing DotA professionally. But it's not the game you might recognize: droids play a different version of it.

In DotA for droids, there are 245 heroes divided into 5 attributes: strength, agility, intellect, universal, and stamina. But what's more interesting is the mode they are playing in: sixdraft. In this mode, as the name suggests, you are given 6 random heroes to pick among the 245. Then, you ban (exclude) 5 of the given heroes in a particular order. The remaining 6-th hero is the hero you will play with.

Droid N4-2A is participating in the tournament. Of course, other droids will be following and even placing bets. During the sixdraft, the audience will see what 5 heroes Droid N4-2A has banned and in what order. Based on that, they can guess which hero was picked (the 6-th hero) and bet some money on that.

Droid N4-2A wants to earn some money from it as well. He wants to collaborate with his friend Droid C-228PO, who will be betting. For 6 random heroes he gets, he wants to find a way to choose particular 5 heroes and ban them in such an order that his friend will be able to guess the 6-th remaining hero.

They want to think of a universal protocol that will allow them to earn money in every game. Help them!

This problem is in **run-twice** format. This means your program will be executed twice for the same test with different inputs.

During the first run, n is 6. It means your program must act as Droid N4-2A. You will be given 6 heroes, and you need to print 5 of them in some order.

During the second run, n is 5. It means your program must act as Droid C-228PO. You will be provided with 5 heroes: your output from the previous run. Note that the order of heroes will be preserved. You need to print the 6-th hero.

Input

The first line contains n, the number of heroes. The value of n is either 5 or 6.

The second line contains n distinct integers a_i ($1 \le a_i \le 245$).

If n is 5, it is guaranteed that a_i are the heroes you printed during the previous run, and their order is preserved.

Output

When n is 6, print 5 integers: the heroes you want to ban.

When n is 5, print a single integer: the remaining 6-th hero.

Interaction Protocol

This problem is tested with an interactor. For that reason please end the line and flush the output after printing. To flush the output, you can use `fflush(stdout)` in C/C++, `System.out.flush()` in Java, or `sys.stdout.flush()` in Python.

Examples

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

Note

The provided examples are the two invocations on the first test.

Problem D. Nomad Camp

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2.5 seconds
Memory limit:	256 mebibytes

On summer vacation, Amir stayed at his grandmother's house, where she told him stories about how nomadic people in ancient times chose pastures for themselves:

There are only n pastures, numbered from 1 to n, and m roads available. Each pasture belongs to one of the four types: қыстау (winter), көктеу (spring), жайлау (summer), and күзеу (autumn).

Each pasture is initially inhabited by people, regardless of the season. When the season changes, from each pasture, people move to the nearest pasture corresponding to the new season. If there are multiple nearest pastures, they choose the pasture with the smallest number. If there is no pasture for the new season, people become sad and stop moving at all.

Now Amir wonders if it would be possible to gather all the people in one place if people could change the season of the year to any other season, as many times as they like.

Input

The first line contains a single integer T ($1 \le T \le 10^4$): the number of test cases. For each test case:

The first line contains two integers n and m ($1 \le n \le 200$, $1 \le m \le \frac{n \cdot (n-1)}{2}$): the number of pastures and the number of roads between them.

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le 4$): the types of pastures.

Each of the next m lines contains three integers, u_i, v_i, and w_i ($1 \le u_i, v_i \le n$, $1 \le w_i \le 10^5$), which mean there is a bidirectional road between pastures u_i and v_i that has length w_i.

It is guaranteed that the sum of n for all test cases does not exceed 10^5.

It is guaranteed that the sum of m for all test cases does not exceed 10^6.

Output

Output T lines, each of which is the answer to the corresponding test case. As the answer, output "`YES`" if it is possible to gather everyone in one place, and "`NO`" otherwise.

You can output the answer in any case (for example, the strings "`yEs`", "`yes`", "`Yes`", and "`YES`" will be recognized as a positive answer).

Example

standard input	*standard output*
2 4 4 1 2 2 4 1 2 5 2 3 100 3 4 8 1 3 11 7 9 3 1 3 2 4 1 2 3 5 7 7 1 1 1 2 7 1 5 1 4 7 10 4 5 10 5 2 11 2 7 3 3 4 10	YES NO

Problem E. Data Structures Master

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Today, Esmaan decided to prove to the world that he is not ordinary. He went to take the exam to become a data structures master. But the first question of the exam stumped him. Help him solve the problem:

You have a sequence of integers $a_1, a_2, \ldots, a_n$. In addition, you have three empty sequences: A, B, and C.

- Let $f(\ell, r)$ be the maximum among the numbers $a_\ell, a_{\ell+1}, \ldots, a_r$.
- Let $g(p_1, p_2, p_3)$ be $f(\min(p_1, p_2, p_3), \max(p_1, p_2, p_3))$.
- Let S be the sum of the values $g(A_i, B_j, C_k)$ for all possible combinations (i, j, k) where $1 \le i \le \text{size}(A)$, $1 \le j \le size(B)$, and $1 \le k \le \text{size}(C)$.

You need to perform q queries of the following type:

- "X *val*": add the value *val* to the end of sequence X.

After each query, output S modulo 998 244 353.

Input

The first line contains two integers n and q ($1 \le n, q \le 10^5$): the number of elements in the sequence and the number of queries.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^5$): the elements of the sequence.

Then follow q lines, each containing a query in the format "X *val*" ($X \in \{\texttt{A}, \texttt{B}, \texttt{C}\}$, $1 \le val \le n$).

Output

After each query, output a line with a single integer: the current value of S modulo 998 244 353.

Examples

standard input	*standard output*
5 5 2 2 9 1 10 A 5 A 1 C 4 B 1 C 5	0 0 0 19 39
10 10 5 6 5 5 10 4 8 9 5 4 C 8 C 8 B 8 B 2 A 7 C 7 C 4 B 4 A 7 B 5	0 0 0 0 38 57 77 117 234 314

Problem F. Poisonous Labyrinth

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

BThero needs to escape from a labyrinth which is represented by a tree with n vertices and $n-1$ edges, where each edge has its own length. Additionally, the labyrinth vertices contain $2m$ poison vials: two vials of each of the m different types of poison.

When BThero enters a vertex for the first time, he immediately drinks all the vials in that vertex. When he ends up in a vertex where he has been before, there are no more vials to drink there.

When BTHero drinks a vial of some type of poison that he did not yet drink, he is poisoned by that type of poison. To cure it, BThero must find and drink the other vial of the same type of poison.

BThero starts his path in vertex s, where he immediately drinks all the vials in that vertex. Then he passes through some vertices until he is no longer poisoned, after which he returns to vertex s and leaves the labyrinth.

It is necessary to find the starting vertex s such that, if BThero starts his path in this vertex, he will have to travel the minimum total distance, provided that he chooses the optimal route.

Input

The first line contains two integers n and m ($2 \le n \le 2 \cdot 10^5$, $1 \le m \le 2 \cdot 10^5$): the number of vertices in the labyrinth and the number of types of poisons.

Each of the next $n-1$ lines contains three integers, u_i, v_i, and w_i ($1 \le u_i, v_i \le n$, $1 \le w_i \le 10^9$) which describe a bidirectional edge between vertices u_i and v_i with length w_i.

Each of the next m lines contains two integers a_j and b_j ($1 \le a_j, b_j \le n$): the two vertices where the vials with poison of type j are located. Note that it is possible that $a_j = b_j$, in which case, when entering the vertex, BThero is poisoned and then cured immediately.

Output

Output a line with a single integer: the minimum distance that BThero will have to travel to cure himself from all poisons if he starts from the optimal vertex.

Examples

standard input	*standard output*
4 2 1 2 1 1 3 10 1 4 100 1 3 2 4	20
5 2 1 2 1 1 3 10 1 4 100 1 5 1000 1 3 2 4	0
7 4 1 2 8 1 3 9 2 4 10 2 5 11 3 6 12 3 7 13 2 3 7 6 2 1 4 5	34

Problem G. Geometry Enjoyer

Input file:	*standard input*
Output file:	*standard output*
Time limit:	7 seconds
Memory limit:	512 mebibytes

Altair was playing with the points on the plane (as usual). At some point, he discovered a new game that he will play with you.

He made a convex polygon with k sides on the two-dimensional plane. The polygon had a really nice property: no pair of sides are parallel. Then he extended every side of the polygon to a line, and found the intersection point for every pair of lines.

Now he gives you the points he got. You should find the initial polygon.

Input

The first line contains one integer n ($1 \le n \le 200$): the number of points.

Each of the next n lines contains four integers, p_x, q_x, p_y, and q_y ($-10^6 \le p_x, p_y \le 10^6$, $1 \le q_x, q_y \le 10^6$): the coordinates of the i-th point. The X coordinate equals p_x/q_x, and the Y coordinate equals p_y/q_y. It is guaranteed that the values p_x and q_x are coprime, and the values p_y and q_y are coprime.

It is guaranteed that the polygon can be uniquely determined by the given points.

Output

The first line of the output should contain one integer k: the size of the polygon.

You can output the vertices of the polygon in any order.

Each of the next k lines should contain four integers, p_x, q_x, p_y, and q_y ($-10^6 \le p_x, p_y \le 10^6$, $1 \le q_x, q_y \le 10^6$): the coordinates of the polygon vertices. The X coordinate equals p_x/q_x, and the Y coordinate equals p_y/q_y. The values p_x and q_x should be coprime, and the values p_y and q_y should be coprime.

Example

standard input	*standard output*
6 1 1 2 1 12 5 24 5 0 1 0 1 3 1 3 1 -3 1 0 1 4 1 0 1	4 0 1 0 1 1 1 2 1 3 1 3 1 4 1 0 1

Problem H. Hocolate Hame

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Azizkhan and Temirulan love Swiss chocolate. Recently they bought a chocolate bar which is a row of n pieces. Each piece has a certain amount of sweetness. Moreover, the sweetness of a piece can be negative.

To divide the chocolate bar in a fair way, they devised some rules for eating the pieces:

- The pieces are numbered $1, 2, \ldots, n$ from left to right.
- Azizkhan will eat the chocolate bar from the left side, and Temirulan from the right side.
- Azizkhan and Temirulan will eat the chocolate in turns.
- Each player eats one or more pieces in his turn.
- Azizkhan eats first, and can eat either 1 or 2 pieces.
- If k pieces were eaten on the previous move, then the current player should eat either k or $k + 1$ pieces.
- They stop eating if it is impossible to make the next move.

Azizkhan and Temirulan are both competitive persons. Each of them wants to consume more sweetness than the other. In other words, each player tries to maximize the difference between the total sweetness of the pieces he ate himself and the total sweetness of the pieces eaten by the opponent. Help them to find the difference between the total sweetness consumed by Azizkhan and Temirulan if both players are super-puper-monstro-smart-optimal persons.

Input

The first line contains an integer n ($1 \le n \le 4000$): the number of pieces in the chocolate bar.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^4 \le a_i \le 10^4$): the sweetness values of the pieces.

Output

Output a line with a single integer: the difference between the total sweetness consumed by Azizkhan and the total sweetness consumed by Temirulan (in this order) if both play optimally.

Examples

standard input	*standard output*
5 1 2 3 4 5	-3
6 -2 5 100 6 3 4	90

Problem I. Lost Table

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Er-Tostik had a table of size $n \times m$ with positive integers. Aldar-Kose decided to prank Er-Tostik and stole the table, but told Er-Tostik the maximum value in each row and column. Aldar-Kose will only return the table if Er-Tostik can tell how many different tables can have these maximum values. As their number can be very large, Aldar-Kose only asks to find this value modulo $10^9 + 7$. Help Er-Tostik to get his table back.

Input

The first line of input contains two integers n and m ($1 \le n, m \le 2 \cdot 10^5$): the dimensions of the table.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$): the maximum values in each row.

The third line contains m integers $b_1, b_2, \ldots, b_m$ ($1 \le b_j \le 10^9$): the maximum values in each column.

Output

Output a line with a single integer: the number of different tables satisfying the conditions. Since the answer can be very large, output it modulo $10^9 + 7$.

Note that, as Aldar-Kose is mischievous, the input might not be consistent with any table at all. In such case, naturally, the correct answer is 0.

Examples

standard input	*standard output*
3 3 2 2 3 2 3 3	89
1 1 1 2	0
5 5 2 2 3 3 3 2 2 2 3 3	49049891
12 13 2 2 2 3 3 4 4 4 4 5 5 5 2 3 3 3 3 4 5 5 5 5 5 5 5	808346164
2 3 2 3 3 1 5	0

Problem J. Team Training

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Today, Amir had another training session with his coach Timo from Kazakhstan.

Today, Timo had $3n$ students at his training session. Each student, indexed as i, had their own level denoted as p_i. It is important to note that all students had different levels.

Usually, Timo divided the students into teams of three participants. However, today he decided to change the system and divide the students into three teams. His approach was as follows: he selected three consecutive students from the list and distributed them among the three teams. The first student was sent to the first team, the second one to the second team, and the third one to the third team. Then he crossed out these three students from the list and repeated the process until all the students were distributed.

The level of a team was determined by the sum of the levels of participants in it. Timo wanted to maximize the level of the first team. If there were multiple options for maximizing the level of the first team, he would maximize the level of the second team. If there were still multiple options, he would maximize the level of the third team.

For example, consider the list of students $[3, 1, 5, 4, 2, 6]$. Suppose Timo would first choose $[1, 5, 4]$, then $[3, 2, 6]$. As a result, the teams would have the following levels: [1+3, 5+2, 4+6] = $[4, 7, 10]$. However, if Timo had chosen $[5, 4, 2]$ first, and then $[3, 1, 6]$, the team levels would have been $[8, 5, 8]$, which is a better distribution according to the criteria.

Find the levels of the teams if Timo divides people into teams optimally.

Input

The first line contains a single integer t ($1 \le t \le 10^5$): the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^5$): the number of students.

The second line contains $3n$ distinct integers $p_1, p_2, \ldots, p_{3n}$ ($1 \le p_i \le 3n$): the levels of the students.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

Output a line with three integers: the levels of the first, second, and third teams in the optimal division.

Example

standard input	*standard output*
2 1 2 3 1 2 3 1 5 4 2 6	2 3 1 8 5 8

Problem K. Reachability in a Matrix

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

You are given a matrix A of size $n \times m$ consisting of **distinct** integers from 1 to $n \cdot m$. The rows of the matrix are numbered from 1 to n, and the columns are numbered from 1 to m. Also, a **positive** integer k is given.

Let us construct a graph consisting of $n \cdot m$ vertices, where the vertices will be the cells of the matrix, labeled as (a, b) ($1 \le a \le n$, $1 \le b \le m$). We will draw a **directed** edge from cell (a, b) to cell (c, d) if both of the following conditions are met:

- The cells are in the same row or column of the matrix. More formally, $a = c$ or $b = d$.
- $A_{a,b} \ge A_{c,d} + k$.

You are given q queries of the form (a, b, c, d). You need to determine whether there exists a path in this graph along the directed edges, starting at vertex (a, b) and ending at vertex (c, d).

Input

The first line of the input file contains three integers, n, m, and k ($1 \le n, m \le 250$, $1 \le k \le n \cdot m$).

Each of the next n lines contains m integers separated by spaces: the values $A_{i,j}$ ($1 \le A_{i,j} \le n \cdot m$). It is guaranteed that all numbers in the matrix are distinct.

The next line contains a single integer q: the number of queries ($1 \le q \le 250\,000$).

Each of the next q lines contains four integers, a_i, b_i, c_i, and d_i: the vertices in the i-th query ($1 \le a_i, c_i \le n$, $1 \le b_i, d_i \le m$, $(a_i, b_i) \ne (c_i, d_i)$).

Output

For each of the q queries, output a line with the word "`Ia`" if a path exists. Otherwise, output a line with the word "`Joq`".

Example

standard input	*standard output*
3 3 2 2 4 6 1 8 3 5 9 7 6 3 2 1 3 3 1 1 1 3 2 1 1 1 3 2 1 3 2 2 3 2 2 3 3	Joq Ia Ia Ia Ia Joq

Note

In the third query, there exist paths $(3, 2) \to (3, 1) \to (1, 1)$ and $(3, 2) \to (1, 2) \to (1, 1)$.

In the fourth query, there exists a path $(1, 3) \to (2, 3) \to (2, 1)$.

Problem L. Bitvzhuh

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Daniyar recently learned a new spell called "Bitvzhuh". Although it is a very high level spell, Daniyar was able to master it completely and unlock its deepest secrets.

"Bitvzhuh", when cast on a set of integers, transforms the set into a new set which contains the XORs of all pairs in the initial set.

Formally, say you have a set $A = \{a_1, a_2, \ldots, a_n\}$ of size n. After one "Bitvzhuh", A turns into the set $\{a_i \oplus a_j \mid 1 \le i < j \le n\}$, where $\oplus$ denotes the bitwise XOR operation.

Given the initial set and the number k, find out if Daniyar can apply "Bitvzhuh" a certain **non-zero** number of times so that the resulting set will contain each integer in the range $[1, 2^k - 1]$.

Input

The first line contains two integers n and k ($3 \le n \le 10^6$, $2 \le k \le 62$): the size of the initial set and the parameter.

The second line contains n distinct integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i < 2^k$): the elements of the initial set.

Output

Print a single line with the word "`Yes`" if the set will contain each integer in the range $[1, 2^k - 1]$ after a certain **non-zero** number of casts of "Bitvzhuh". Otherwise, print a single line with the word "`No`".

Examples

standard input	*standard output*
4 3 1 2 3 4	Yes
4 3 1 2 4 7	No

Note

In the first example, the answer is achieved after two casts:

$\{1, 2, 3, 4\} \to \{1, 2, 3, 5, 6, 7\} \to \{1, 2, 3, 4, 5, 6, 7\}$.

In the second example, the first cast turns the set $\{1, 2, 4, 7\}$ into $\{3, 5, 6\}$, and it never changes after.