Problem A. The Story of Company R

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

How many people can be as successful as wcysai?

Company R is a money-printing company founded by wcysai, with an annual revenue of over 100 million. There are n employees working at Company R, and each employee lives in a house. Some employees' houses are connected by bidirectional roads. It is guaranteed that starting from any employee's house, one can reach any other employee's house through the roads.

Employees are divided into two types: managers (type A) and workers (type B). A manager works a hours per day, while a worker works b hours per day. Obviously, workers must work longer than managers, so $a < b$. Each employee has a starting work time s_i. If this employee needs to work m hours ($m = a$ or $m = b$), then their ending time is $t_i = s_i + m$.

However, employees never want to be the most overworked one. Therefore, when they start work, they will check whether any of their neighbors (that is, employees whose houses are directly connected by roads) have already started working. Similarly, when they finish work, they will check whether any of their neighbors have not yet finished working. If both observations are positive, they will feel satisfied. Formally, an employee is satisfied if and only if there exists a neighbor whose starting time is **strictly** earlier than theirs, and there exists a neighbor whose ending time is **strictly** later than theirs (these two neighbors may be the same person or different people). Otherwise, the employee will be dissatisfied because they feel they are the most overworked one.

Now, as wcysai's capable assistant, you may freely arrange the starting time of each employee. You want to schedule all employees reasonably so that the number of dissatisfied people is minimized. Note that you may assign starting times to **any non-negative real numbers** (see the output format for details).

Input

The first line contains an integer T ($1 \le T \le 10^5$), the number of test cases.

For each test case, the first line contains four non-negative integers n, m, a, and b ($1 \le n, m \le 5 \cdot 10^5$, $1 \le a < b \le 10^6$), representing the number of employees, the number of roads, the working hours of managers, and the working hours of workers, respectively.

The next line contains a string of length n consisting only of "A" and "B". The i-th character represents the type of the i-th employee.

The next m lines each contain two non-negative integers u, v ($1 \le u, v \le n$), representing a road.

It is guaranteed that if the employees' houses are regarded as vertices and the roads as edges, the graph has no multiple edges or self-loops, and is connected.

It is guaranteed that the sum of n and the sum of m over all test cases do not exceed $5 \cdot 10^5$.

Output

For each test case, output one line containing n non-negative integers s'_i ($0 \le s'_i \le 10^{18}$). Specifically, let s_i denote the time when the i-th employee starts working, then $s_i = \frac{s'_i}{10^6}$. If there are multiple optimal schedules that minimize the number of dissatisfied people, output any of them.

Example

standard input	standard output
5 2 1 1 5 AA 1 2 3 3 1 5 ABA 1 2 1 3 2 3 5 4 2 4 BABBB 1 2 1 5 2 3 2 4 6 9 1 5 BABBAB 1 2 1 3 1 4 2 6 3 4 3 5 3 6 4 5 5 6 7 7 2 5 BBAAABB 1 2 1 3 1 5 2 4 3 6 3 7 6 7	0 1 1 0 2 2 1 0 3 4 2 3 1 4 4000005 0 3000003 3000004 3000002 3000005 3000006 1 0

Problem B. Cloud-Ascending Platform

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

Just like many things in the world, the title of this problem has no meaning.

Given an undirected graph with n vertices and m edges, the edges are numbered from 1 to m in order, and each edge has a weight c_i. You now need to answer q queries, each in the following form:

- If only the edges with indices in $[l, r]$ are kept, what is the sum of the weights of all bridge edges in the graph modulo 2^{64}?

You need to answer each query.

You may want to know the definition of a bridge: an edge is called a bridge if and only if removing it increases the number of connected components in the graph.

Input

The first line contains n, m, q ($1 \le n \le 500, 1 \le m, q \le 10^5$), representing the number of vertices, edges, and queries in the graph.

The next m lines each contain three integers u, v, c ($1 \le u, v \le n$, $0 \le c < 2^{64}$), representing an edge in the graph.

The next q lines each contain two positive integers l, r ($1 \le l \le r \le m$), representing a query.

It is guaranteed that the graph contains no self-loops, but it may contain multiple edges.

Output

For each query, output one line containing an integer representing the answer.

Example

standard input	standard output
5 6 6 1 2 1 2 3 2 2 4 4 2 5 8 1 3 16 4 5 32 1 3 2 4 3 5 4 6 2 6 1 6	7 14 28 56 18 0

Problem C. PalindromemordnilaP

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Life is always full of uncertainties. For example, you might think from the problem title that this is a string problem, but actually it is not.

You are given a string s and a character c. Each time, you randomly choose a position i that has **not been chosen before**, and change s_i to c (if $s_i = c$, it still counts as one operation). You repeat this operation until s becomes a palindrome, and then stop. You want to know the expected number of operations needed.

In particular, if s is already a palindrome at the beginning, the answer is 0.

Input

The first line contains a string s ($1 \leq |s| \leq 250000$), guaranteed to consist only of lowercase English letters.

The second line contains a lowercase English letter c.

Output

Output one integer on a single line, representing the expected value modulo 998244353.

Examples

standard input	standard output
abcab a	4
abcdcbd a	177465668
aba z	0

Problem D. GG

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

"If you can't solve a problem, just switch to another one. What if this one is the anti-AK task?"

Given three integers n, a, b.

Consider all intervals $[l, r]$ such that $1 \le l < r \le n$. You need to choose some of them (possibly none), satisfying:

- All chosen intervals are contained in $[1, n]$;
- All endpoints of the chosen intervals are pairwise distinct. In other words, if the chosen intervals are $[l_1, r_1], [l_2, r_2], \ldots, [l_k, r_k]$, then the $2k$ numbers $l_1, r_1, l_2, r_2, \ldots, l_k, r_k$ are pairwise distinct;
- There exist a chosen intervals $[l_1, r_1], [l_2, r_2], \ldots, [l_a, r_a]$ such that $l_1 < l_2 < \cdots < l_a$ and $r_a < r_{a-1} < \cdots < r_1$;
- There exist b chosen intervals $[l_1, r_1], [l_2, r_2], \ldots, [l_b, r_b]$ such that $l_1 < l_2 < \cdots < l_b$ and $r_1 < r_2 < \cdots < r_b$.

Find the number of valid ways to choose the intervals. Since the answer may be very large, output it modulo $10^9 + 7$.

"A wise decision"

Input

The input consists of a single line containing three integers n, a, b ($2 \le n \le 100$, $0 \le a, b \le n$).

Output

Output a single integer, the number of valid interval-selection schemes modulo $10^9 + 7$.

Examples

standard input	standard output
2 0 0	2
2 0 1	1
5 1 2	10
9 2 3	504
100 14 22	36993304

,

Problem E. Is this a problem?

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Of course, this is a problem.

This is an interactive problem. You are given an unknown undirected graph with n vertices and m edges, with no multiple edges or self-loops. Each time, you may ask the interaction library the following question:

- Given a vertex set $S \subseteq \{1, 2, \ldots, n\}$, ask whether this vertex set is an independent set.

Here, an independent set means that there is no edge between any two distinct vertices in the set S.

Your goal is to determine the entire edge set of the graph using no more than $M = n + m(2 + \lceil \log_2 n \rceil)$ queries.

Note that you do not know the value of m in advance, and the interaction library is **adaptive**, i.e. the graph is not fixed from the beginning, but will be adjusted dynamically according to your queries.

Input

The first line contains an integer T ($1 \leq T \leq 100$), denoting the number of test cases.

For each test case, one line contains an integer n ($1 \leq n \leq 500$), denoting the number of vertices of the unknown undirected graph.

It is guaranteed that the sum of n over all test cases does not exceed 500, $0 \leq m \leq 500$, and the sum of m does not exceed 500.

Interaction Protocol

When you need to ask a query, output one line: $?kv_1v_2...v_k$

where k denotes the size of the vertex set, and $v_1, v_2, \ldots, v_k$ are the indices of the vertices in the set you are querying, all pairwise distinct.

The interaction library will return an integer: `1` if the vertex set is an independent set; `0` otherwise.

When you have determined all edges of the graph, output one line: $!mu_1v_1u_2v_2...u_mv_m$

where m is the number of edges in the undirected graph, and (u_i, v_i) represents an edge in the graph. You may output the edges in any order.

After that, if there are no remaining test cases, your program must terminate immediately; otherwise, it should proceed to the next test case.

Example

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

Note

The sample demonstrates an interactive process of querying whether each edge exists individually.

Problem F. Tired

Input file:	standard input
Output file:	standard output
Time limit:	2 seconds
Memory limit:	1024 megabytes

I'm tired of writing, I don't want to write anymore.

"This is called 'tired'."

Given a sequence $a_1, a_2 \dots a_n$ of length n, each position i has a modification cost c_i. For a position, you can pay the cost c_i to change a_i into any value. You want to minimize the total cost so that every prefix sum of the sequence is not divisible by r.

Formally, let $s_i = \sum_{j=1}^{i} a_j$. You need to make sure that for the modified sequence, for all $1 \le i \le n$, $s_i \bmod r \neq 0$.

Input

The first line contains the number of test cases T ($1 \le T \le 10^5$).

For each test case, the first line contains two positive integers n, r ($1 \le n \le 5 \cdot 10^5$, $2 \le r \le 10^9$).

The next line contains n non-negative integers a_i ($0 \le a_i < r$), representing the sequence.

The next line contains n non-negative integers c_i ($0 \le c_i \le 10^9$), representing the modification costs.

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

Output

For each test case, output one integer in a single line, representing the minimum cost.

Example

standard input	standard output
5 3 3 2 1 2 3 2 1 4 2 0 1 0 0 2 1 3 1 5 3 2 1 1 0 2 3 2 4 1 5 6 3 0 2 1 1 2 0 6 2 4 5 5 1 7 4 1 2 3 0 1 2 3 3 4 1 7 5 2 3	2 3 2 10 2

Problem G. Boring Constructive Problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

The most boring constructive problem in the world is to read n, m and output a matrix satisfying conditions 1,2,3,4,5,6,7,8.

Given four integers a, b, n, s ($a \neq b$). You need to construct a sequence c of length n such that:

- For every $1 \leq i \leq n$, $c_i \in \{a, b\}$;
- There does not exist a contiguous subarray $c_l, c_{l+1}, \ldots, c_r$ whose sum is exactly s.

More formally, there do not exist integers (l, r) satisfying $1 \leq l \leq r \leq n$ such that $c_l + c_{l+1} + \cdots + c_r = s$.

For each test case, determine whether such a sequence exists:

- If it exists, output `YES`, and output any valid sequence;
- Otherwise, output `NO`.

Input

The first line contains an integer T ($1 \leq T \leq 10^5$), denoting the number of test cases.

Each of the next T lines contains four integers a, b, n, s ($1 \leq a, b, s \leq 10^9$, $1 \leq n \leq 5 \cdot 10^5$, $a \neq b$).

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

Output

For each test case:

- If there is no solution, output a line containing `NO`;
- If there is a solution, first output a line containing `YES`, then output a line containing n integers representing the sequence you constructed.

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

Example

standard input	standard output
3 2 4 6 8 2 5 5 9 3 4 3 7	NO YES 2 5 5 5 2 YES 4 4 4

Problem H. ucup-team7610

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	1024 megabytes

Why are you not doing Universal Cup?

If you have ever had a ucup account, then you will know that the format of a ucup account is 'ucup-teamXXXX', where 'XXXX' is a number with **at least three digits** and no leading zeros.

Now given several account names, please determine whether they match the ucup account format.

Input

The first line contains the number of test cases T ($1 \le T \le 10^5$).

For each test case, one line contains a string s ($1 \le |s| \le 50$) consisting only of lowercase letters, digits, and the hyphen '-', representing an account name.

Output

For each test case, output one line 'YES' or 'NO', indicating whether it matches the ucup format.

Example

standard input	standard output
5 abcdeabcdeabc ucup-team7610 ucup-team22 ucup-team01234 ucup-team123456789123456789	NO YES NO NO YES

Note

Will we ever live to see the day when 'ucup-team123456789123456789' gets registered?

Problem I. Hikoutei

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

"For problems you don't know how to solve, just submit something random. Who knows, maybe it will pass."

Given an $n \times m$ matrix of numbers, you may arbitrarily rearrange the numbers in each row so that the sum of the minimum values of all columns is maximized.

Formally, let the rearranged matrix be A. Maximize $\sum_{j=1}^{m}(\min_{i=1}^{n} A_{i,j})$.

You need to output this maximum value.

"More often than not, things going against our wishes is simply the norm in life."

Input

The first line of the input contains the number of test cases T ($1 \le T \le 10^5$).

For each test case, the first line contains two positive integers n, m ($1 \le n, m \le 1000$), representing the size of the matrix.

The next n lines each contain m integers, representing the numbers $A_{i,j}$ in the matrix ($0 \le A_{i,j} \le 10^9$).

It is guaranteed that the sum of $n \cdot m$ over all test cases does not exceed 10^6.

Output

For each test case, output a single integer on one line representing the answer.

Example

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

Problem J. Ballon Problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

A contest cannot be without a data structure problem, even if it is only a ballon problem.

You are given a matrix M. Let the element in the i-th row and j-th column of the matrix be $M_{i,j}$, and let the number of rows and columns be n and m, respectively. Initially, $n = m = 1$, and $M_{1,1} = c$.

You need to perform q operations, each of which belongs to one of the following three types:

- `1 x y` ($0 \le x \le n, 1 \le y \le 10^9$) insert a row with value y between the x-th row and the $(x+1)$-th row; after this operation, n increases by 1;
- `2 x y` ($0 \le x \le m, 1 \le y \le 10^9$) insert a column with value y between the x-th column and the $(x+1)$-th column; after this operation, m increases by 1;
- `3 x y` ($1 \le x \le n, 1 \le y \le m$) query the value of $M_{x,y}$.

Input

The first line of input contains two integers q, c ($1 \le q \le 5 \cdot 10^5, 1 \le c \le 10^9$).

The following q lines each describe one operation.

The three integers opt, x, y at the beginning of each line ($opt \in \{1, 2, 3\}$) have the meaning described in the statement above.

Output

For each query operation, output one line containing an integer representing the answer.

Example

standard input	standard output
6 1 1 0 2 3 2 1 2 1 3 1 1 4 3 2 2 3 3 2	1 4 3

Problem K. Soul Link

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

You are playing an adventure game. As a mage, you need to use magic to deal with several enemies in front of you. Each enemy has a level, which is an integer between 1 and n. An enemy of level i has 2^{i-1} health points.

You can use two kinds of spells:

- Soul Link: consumes m mana and applies the "Soul Link"effect to one enemy. This effect lasts until that enemy dies.
- Attack: consumes 1 mana and deals 1 damage to all enemies with the Soul Link effect, i.e. decreases their health by 1. Once an enemy's health drops to 0, that enemy dies, and you gain m mana as a reward.

Among the enemies in front of you, there are a_i enemies of level i. You may use the two spells any number of times, but at any time your mana cannot be less than 0. What is the minimum initial mana you need in order to defeat all enemies?

Input

The first line contains an integer T ($1 \le T \le 500$), denoting the number of test cases.

For each test case, the first line contains two integers n, m ($1 \le n \le 30, 1 \le m \le 10^{18}$), representing the maximum enemy level and the mana cost of Soul Link, respectively.

The second line contains n positive integers. The i-th integer a_i denotes the number of enemies of level i.

For each test case, it is guaranteed that $0 \le a_i \le 10^9$ and $\sum_{i=1}^{n} a_i > 0$.

Output

For each test case, output one line containing an integer, denoting the minimum initial mana required to defeat all enemies.

Example

standard input	standard output
2 5 7 5 2 4 1 2 6 4 1 1 4 5 1 4	34 48

Note

Explanation for the first test case:

Initially, there are 14 enemies, with health values $1, 1, 1, 1, 1, 2, 2, 4, 4, 4, 4, 8, 16, 16$.

With 34 mana initially, first apply Soul Link to the enemies with health $1, 1, 1, 1$, leaving 6 mana.

Attack once, and the enemies' health becomes $0, 0, 0, 0, 1, 2, 2, 4, 4, 4, 4, 8, 16, 16$. Four enemies die, leaving 33 mana.

Apply Soul Link to the enemies with health $1, 2, 2, 4$, leaving 5 mana.

Attack once, and the enemies' health becomes $0, 1, 1, 3, 4, 4, 4, 8, 16, 16$. One enemy dies, leaving 11 mana.

Apply Soul Link to the enemy with health 4, leaving 4 mana.

Attack once, and the enemies' health becomes $0, 0, 2, 3, 4, 4, 8, 16, 16$. Two enemies die, leaving 17 mana.

Apply Soul Link to the enemies with health $4, 16$, leaving 3 mana.

Attack twice, and the enemies' health becomes $0, 1, 2, 4, 8, 14, 16$. One enemy dies, leaving 8 mana.

Apply Soul Link to the enemy with health 8, leaving 1 mana.

Attack once, and the enemies' health becomes $0, 1, 4, 7, 13, 16$. One enemy dies, leaving 7 mana.

Attack once, and the enemies' health becomes $0, 4, 6, 12, 16$. One enemy dies, leaving 13 mana.

Apply Soul Link to the enemy with health 4, leaving 6 mana.

Attack four times, and the enemies' health becomes $0, 2, 8, 16$. One enemy dies, leaving 9 mana.

Apply Soul Link to the enemy with health 16, leaving 2 mana.

Attack twice, and the enemies' health becomes $0, 6, 14$. One enemy dies, leaving 7 mana.

Attack six times, and the enemies' health becomes $0, 8$. One enemy dies, leaving 8 mana.

Attack eight times, and all enemies die, leaving 0 mana.

Problem L. Marcel Adventure

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

You have a cute Bangboo, and it is participating in Marcel's Great Adventure.

This Bangboo moves in three-dimensional space. Its starting point is (S_x, S_y, S_z), and its destination is (T_x, T_y, T_z). You may regard it as a unit cube centered at (S_x, S_y, S_z). There are n obstacles in the three-dimensional space, each of which is a unit cube centered at (x_i, y_i, z_i). It is guaranteed that the coordinates of the starting point, the destination, and all obstacles are pairwise distinct, and that there must be obstacles at $(S_x, S_y, S_z - 1)$ and $(T_x, T_y, T_z - 1)$.

The Bangboo can teleport. At each step, it has m teleportation methods. Suppose it is currently at (x, y, z). Then the i-th teleportation method sends it to $(x + a_i, y + b_i, z + c_i)$, after which it undergoes free fall. Formally, it will teleport to $(x + a_i, y + b_i, d)$, where d is the maximum value satisfying $d \le z + c_i$ and $(x + a_i, y + b_i, d - 1)$ is an obstacle. If there is an obstacle at $(x + a_i, y + b_i, z + c_i)$, or if no such d exists, then this teleportation cannot be performed.

Your goal is to make it reach the destination from the starting point using the minimum number of steps. If it is impossible to reach the destination, output -1.

Input

The first line contains an integer T ($1 \le T \le 500$), denoting the number of test cases.

For each test case, the first line contains two integers n, m ($2 \le n \le 10^3, 1 \le m \le 10^3$), denoting the number of obstacles and the number of teleportation methods available to the Bangboo.

The second line contains six integers $S_x, S_y, S_z, T_x, T_y, T_z$ ($0 \le S_x, S_y, S_z, T_x, T_y, T_z \le 10^9$), denoting the starting point and the destination of the Bangboo.

The next n lines each contain three integers x_i, y_i, z_i ($0 \le x_i, y_i, z_i \le 10^9$), denoting the coordinates of the i-th obstacle. It is guaranteed that there must be obstacles at $(S_x, S_y, S_z - 1)$ and $(T_x, T_y, T_z - 1)$.

The next m lines each contain three integers a_i, b_i, c_i ($|a_i|, |b_i|, |c_i| \le 10^9$), denoting the i-th teleportation method.

For each test case, it is guaranteed that all input coordinates are pairwise distinct, and all teleportation methods are pairwise distinct.

It is guaranteed that the sum of n over all test cases does not exceed 10^3, and the sum of m over all test cases does not exceed 10^3.

Output

For each test case, output a single integer in one line, denoting the minimum number of steps required to reach the destination from the starting point. If it is impossible to reach the destination, output -1.

Example

standard input	standard output
2 6 4 3 3 3 5 5 5 3 3 2 5 5 4 5 5 7 4 5 5 2 3 2 4 1 1 -1 0 3 2 -2 -1 1 4 9 1 0 0 3 2 0 0 1 1 1 1 0 0 0 1 1 0 1 1 2 1 1 1 1 1 2	5 -1

Problem M. Rounddog

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

"There must be an end to every beginning"

For a sequence $A = (a_1, a_2, \ldots, a_m)$, define:

- the length of the strict longest increasing subsequence:

$$\mathrm{LIS}(A) = \max\{k \mid \exists 1 \le i_1 < i_2 < \cdots < i_k \le m,\ a_{i_1} < a_{i_2} < \cdots < a_{i_k}\}.$$

- the length of the strict longest decreasing subsequence:

$$\mathrm{LDS}(A) = \max\{k \mid \exists 1 \le i_1 < i_2 < \cdots < i_k \le m,\ a_{i_1} > a_{i_2} > \cdots > a_{i_k}\}.$$

For any sequence A, define $f(A)$ as the maximum integer k satisfying the following conditions:

There exist k real numbers $x_1, x_2, \ldots, x_k$. After appending them to the end of A in order, we obtain a new sequence $B = (a_1, a_2, \ldots, a_m, x_1, x_2, \ldots, x_k)$, and the following hold:

1. all elements in sequence B are pairwise distinct;
2. $\mathrm{LIS}(B) = \mathrm{LIS}(A)$;
3. $\mathrm{LDS}(B) = \mathrm{LDS}(A)$.

Now given a permutation $P = (P_1, P_2, \ldots, P_n)$ of length n, for each prefix $P^{(i)} = (P_1, P_2, \ldots, P_i)$, you need to compute $f\left(P^{(i)}\right)$ and output all answers.

"One day I will return to your side"

Input

The first line contains an integer T ($1 \le T \le 10^5$), the number of test cases.

For each test case:

- The first line contains an integer n ($1 \le n \le 2 \cdot 10^5$);
- The second line contains n integers $P_1, P_2, \ldots, P_n$ ($1 \le P_i \le n$), representing a permutation of length n.

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

Output

For each test case, output one line containing n integers representing the answers.

Example

standard input	standard output
5 2 1 2 3 1 3 2 4 3 4 1 2 5 2 5 1 3 4 6 3 6 2 1 4 5	0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 2 1 2