Problem A. Super-palindrome

You are given a string that is consisted of lowercase English alphabet. You are supposed to change it into a super-palindrome string in minimum steps. You can change one character in string to another letter per step.

A string is called a super-palindrome string if all its substrings with an odd length are palindrome strings. That is, for a string s, if its substring $s_{i \cdots j}$ satisfies $j - i + 1$ is odd then $s_{i+k} = s_{j-k}$ for $k = 0, 1, \cdots, j - i + 1$.

Input

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

For each test case, the only line contains a string, which consists of only lowercase letters. It is guaranteed that the length of string satisfies $1 \leq |s| \leq 100$.

Output

For each test case, print one line with an integer refers to the minimum steps to take.

Example

standard input	standard output
3 ncncn aaaaba aaaabb	0 1 2

Explanation

For second test case `aaaaba`, just change letter `b` to `a` in one step.

Problem B. Master of Phi

You are given an integer n. Please output the answer of $\sum_{d|n} \varphi(d) \times \frac{n}{d}$ modulo 998244353. n is represented in the form of factorization.

$\varphi(n)$ is Euler's totient function, and it is defined more formally as the number of integers k in the interval $1 \leq k \leq n$ for which the greatest common divisor $\gcd(n, k)$ is equal to 1.

For example, the totatives of $n = 9$ are the six numbers 1, 2, 4, 5, 7 and 8. They are all co-prime to 9, but the other three numbers in this interval, 3, 6, and 9 are not, because $\gcd(9, 3) = \gcd(9, 6) = 3$ and $\gcd(9, 9) = 9$. Therefore, $\varphi(9) = 6$. As another example, $\varphi(1) = 1$ since for $n = 1$ the only integer in the interval from 1 to n is 1 itself, and $\gcd(1, 1) = 1$.

And there are several formulas for computing $\varphi(n)$, for example, Euler's product formula states like:

$$\varphi(n) = n \prod_{p|n} \left(1 - \frac{1}{p}\right),$$

where the product is all the distinct prime numbers (p in the formula) dividing n.

Input

The first line contains an integer T ($1 \leq T \leq 20$) representing the number of test cases.

For each test case, the first line contains an integer m ($1 \leq m \leq 20$) is the number of prime factors.

The following m lines each contains two integers p_i and q_i ($2 \leq p_i \leq 10^8$, $1 \leq q_i \leq 10^8$) describing that n contains the factor ${p_i}^{q_i}$, in other words, $n = \prod_{i=1}^{m} {p_i}^{q_i}$. It is guaranteed that all p_i are prime numbers and different from each other.

Output

For each test case, print the the answer modulo 998244353 in one line.

Example

standard input	standard output
2 2 2 1 3 1 2 2 2 3 2	15 168

Explanation

For first test case, $n = 2^1 * 3^1 = 6$, and the answer is $(\varphi(1) * n/1 + \varphi(2) * n/2 + \varphi(3) * n/3 + \varphi(6) * n/6) \mod 998244353 = (6 + 3 + 4 + 2) \mod 998244353 = 15$.

Problem C. Hakase and Nano

Hakase and Nano are playing an ancient pebble game (pebble is a kind of rock). There are n packs of pebbles, and the i-th pack contains a_i pebbles. They take turns to pick up pebbles. In each turn, they can choose a pack arbitrarily and pick up at least one pebble in this pack. The person who takes the last pebble wins.

This time, Hakase cheats. In each turn, she must pick pebbles following the rules twice continuously. Suppose both players play optimally, can you tell whether Hakase will win?

Input

The first line contains an integer T ($1 \le T \le 20$) representing the number of test cases.

For each test case, the first line of description contains two integers $n(1 \le n \le 10^6)$ and d ($d = 1$ or $d = 2$). If $d = 1$, Hakase takes first and if $d = 2$, Nano takes first. n represents the number of pebble packs.

The second line contains n integers, the i-th integer a_i ($1 \le a_i \le 10^9$) represents the number of pebbles in the i-th pebble pack.

Output

For each test case, print "`Yes`" or "`No`" in one line. If Hakase can win, print "`Yes`", otherwise, print "`No`".

Example

standard input	standard output
2 3 1 1 1 2 3 2 1 1 2	Yes No

Problem D. Master of Random

Hakase provides Nano with a problem. There is a rooted tree with values on nodes. For each query, you are asked to calculate the sum of the values in the subtree. However, Nano is a rookie so she decides to guess the answer. She has known how the data generator works: it identifies the nodes with labels from 0 to $n-1$ and then visits them one by one. For each i $(1 \leq i \leq n)$, the generator selects a node whose label is smaller than i to be its father. The pseudocode is like this:

```
for i = 1 to n - 1:
    father[i] = random(0, i - 1);
```

where `random(a, b)` randomly generates a uniformly distributed random integer in range [a, b].

Knowing n and the value of the i-th node a_i, Nano decides to randomly choose a subtree and sum up all of the values in the subtree as the answer. Now Hakase wants to know what the expectation of the answer is. Can you help her?

Input

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

For each test case, the first line contains an integer n $(1 \leq n \leq 100000)$, the number of the nodes in the rooted tree.

The second line contains n integers $a_0, a_1, ..., a_{n-1}$ $(1 \leq a_i \leq 100000)$ represent the values of nodes.

Output

It can be proven that the answer equals to an irreducible fraction p/q. For each test case, print $p * q^{-1}$ mod 998244353 in one line. q^{-1} is the inverse of q under module number 998244353.

Example

standard input	standard output
2 2 1 1 3 1 2 3	499122178 166374063

Explanation

The shape of the tree in the first test case is unique. The father of node 1 is 0. It is possible to choose node 0 or 1 with equal possibility. The sum of the subtree with 0 as the root is 2 while the sum of the subtree with 1 as the root is 1. So the expectation is $(2+1)/2 = 3/2$. The output is $3 * 2^{-1} \mod 998244353 = 400122178$.

There are two possible shapes in the second test case, node 1's father destines to be 0, but node 2's father might be node 0 or node 1. Both conditions are equally possible.

If node 2's father is node 0, we randomly choose a node. The sum of the subtree with node 0 as the root is 6. The sum of the subtree with node 1 as the root is 2. The sum of the subtree with node 2 as the root is 3.

If node 2's father is node 1, we randomly choose a node. The sum of the subtree with node 0 as the root is 6. The sum of the subtree with node 1 as the root is 5. The sum of the subtree with node 2 as the root is 3.

So the expectation is $(6 + 2 + 3 + 6 + 5 + 3)/6 = 25/6$. The output is $25 * 6^{-1}\ mod\ 998244353 = 166374063$.

Problem E. Master of Subgraph

You are given a tree with n nodes. The weight of the i-th node is w_i. Given a positive integer m, now you need to judge that for every integer i in $[1, m]$ whether there exists a connected subgraph which the sum of the weights of all nodes is equal to i.

Input

The first line contains an integer T $(1 \leq T \leq 15)$ representing the number of test cases.

For each test case, the first line contains two integers n $(1 \leq n \leq 3000)$ and m $(1 \leq m \leq 100000)$, which are mentioned above.

The following $n-1$ lines each contains two integers u_i and v_i $(1 \leq u_i, v_i \leq n)$. It describes an edge between node u_i and node v_i.

The following n lines each contains an integer w_i $(0 \leq w_i \leq 100000)$ represents the weight of the i-th node.

It is guaranteed that the input graph is a tree.

Output

For each test case, print a string only contains `0` and `1`, and the length of the string is equal to m. If there is a connected subgraph which the sum of the weights of its nodes is equal to i, the i-th letter of string is `1` otherwise `0`.

Example

standard input	standard output
2 4 10 1 2 2 3 3 4 3 2 7 5 6 10 1 2 1 3 2 5 3 4 3 6 1 3 5 7 9 11	0110101010 1011111010

Problem F. Hearthrock

You are playing the card game "Hearthrock". The rules are described as follows:

You and your enemy both control a hero with some heart points(HP), your hero has H_m heart points, and your enemy's hero has H_e heart points. At the end of any player's move, if any player's hero's HP is less or equal to 0, the game is over and the other player wins.

Players can summon minions and release spells during the game.

The minion will lose the same HP as the attack it received. If the minion's HP is less or equal to 0, the minion will die.

In each turn, you move first and then the other player. Through observation, you find out that the enemy might be a bot. Pathetically, you are still in woeful situation. You have no minion while your enemy have 3 alive minions.

Every minion have three attributes: a_i, b_i, c_i, meaning the attack ability, current HP and the superior limit of HP.

You realized in each turn your enemy will move unchangeably: he will command all of his minions who are still alive to attack you, causing damage equal to the sum of their a_i. After that, he will summon all of the minions you killed in this turn. When the minions are summoned, their current HP is equal to the superior limit which means $b_i = c_i$

Unluckily, you have no more other cards in your deck, so you will obtain no new cards. Luckily, the game has no fatigue value so you can ignore it, in other word, you will not receive damage cause lack of cards in your deck.

Now you have up to eight cards within three different types cards:

1. the first kind will cause x_i damage to all of the enemy's minions and then obtain y_i points Spell Power in this turn.
2. the second kind will cause x_i damage to the hero of the enemy's and then obtain y_i points Spell Power in this turn.
3. cause x_i damage to one enemy's character which might be the hero or a minion.

Spell Power: if you have h points of Spell Power, you can cause extra h damage. It means that if you cause t damage, the factual damage is $t + h$. Your Spell Power will be 0 at the beginning of each turn.

In each turn, you will have 10 mana crystals. The i-th card consumes $cost_i$ ($cost_i \geq 2$) mana crystals. The mana crystal can't be accumulated to next turn. In each turn you can play any number of cards under the restriction of mana crystals.

During the game, the cards can't be used more than once, in other word, if you play one card in some turn, you can not use it later.

In order to avoid being found well-matched with the bot, you need to find whether you can defeat your enemy within three turns.

Input

The first line contains an integer T ($T = 5$) representing the number of test cases.

For each test case, the first line contains an integer n ($1 \le n \le 8$), the number of your cards, labeled from 1 to n.

The second line contains two integers H_m and H_e, your hero's HP and your enemy's hero's HP.

The following three lines describe three enemy's minions, containing a_i, b_i, c_i which are described above.

The next n lines describe n cards. For type 1 and type 2, the cards have four integers t_i, c_i, x_i, y_i. t_i means the type of the card and $cost_i$, x_i with y_i were described above. For type 3, the card will contain three integers t_i, $cost_i$, x_i.

It's guaranteed that all numbers will not exceed 100, and all input numbers will be non-negative integer.

Output

For each test case, if you can't win within 3 turns, print "`No`".

Otherwise, print "`Yes`" and print three blocks for three turns.

For each block, the first line contains an integer k, representing that you use k cards in this turn. **If $k = 0$, you should not print the next 2 lines.**

The second line contains k integers. $Card_i$ represents the i-th cards you used in this turn.

The third line contains k integers. i-th integer $Targ_i$ represents the target of the i-th card. If the type of the card is 1 or 2, print "`-1`". Otherwise, if the type is 3, print "`0`" to attack hero or an integer in $\{1, 2, 3\}$ to attack the $targ_i$-th minion.

Example

standard input	standard output
2 3 30 30 10 20 20 10 20 20 10 20 20 1 5 19 0 2 5 5 5 3 5 20 6 1 25 1 10 10 1 10 10 1 10 10 1 3 3 0 2 3 0 7 3 3 1 2 3 1 5 2 3 1 4 2 3 1 3	Yes 2 2 3 -1 0 0 0 Yes 3 2 1 3 -1 -1 0 3 4 5 6 -1 -1 -1 0

Problem G. Marriage

There are n families in S City. The i-th family has a_i boys and b_i girls (a_i and b_i may be 0). The total number of boys in S City is equal to girls.

S City government is going to assign the marriage. Each boy has exactly one girl to be his wife and each girl has exactly one boy to be her husband. But the boy and girl from one family must not be a couple.

You are required to answer how many different arrangements are there in total. **As the answer could be very large, find it modulo 998244353.** Two arrangements are considered different if there exists a boy matches different girls.

Input

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

For each test case, the first line contains an integer n ($1 \leq n \leq 100000$) which is the number of families.

The following n lines each contains two integers a_i and b_i ($0 \leq a_i, b_i \leq 100000$). It is guaranteed $0 < \sum a_i = \sum b_i \leq 100000$.

Output

For each test case, print the answer in one line. If there is no possible arrangement, just print 0.

Example

standard input	standard output
4 1 3 3 2 0 2 2 0 4 1 1 1 1 1 1 1 1 6 0 2 1 0 1 2 0 0 2 2 2 0	0 2 9 216

Problem H. Master of Connected Component

You have two trees T_a and T_b, and they both have n nodes. All n nodes on each tree are respectively labelled by integers in ranges $[1, n]$. The shape and label scheme of these two trees may be different. And you have a graph G with m nodes and all m nodes on grape G are respectively labelled by integers in ranges $[1, m]$. There are no edges in graph G in the beginning, but every node on trees stores an edge of graph G.

For each index i, it corresponds to node i in tree T_a, and node i in tree T_b. There is a path from node i to the root of tree T_a, and there is a path from node i to the root of tree T_b. Each path contains some nodes on trees, and each node stores an edge in graph G. All the nodes in both paths store some edges in graph G. For each index i, we temporarily add all these edges to graph G and construct a new graph G'. Now you are supposed to answer how many connected components are there in G'. Of course, one single node with no edge is seemed as a connected component.

It is guaranteed that the root of each tree is labelled by number 1.

Input

The first line contains an integer T $(1 \leq T \leq 25)$ representing the number of test cases.

For each test case, the first line contains two integers n $(1 \leq n \leq 10000)$, m $(1 \leq m \leq 10000)$, representing the number of nodes of each tree and the number of nodes of graph G.

Next $n + n - 1$ lines describe tree T_a.

The following n lines each contains two integers u_i, v_i, representing that the node with lable i of tree T_a refers to the undirected edge (u_i, v_i) in graph G.

The following $n - 1$ lines each contains two integers u_i, v_i, representing an edge (u_i, v_i) in tree T_a, and the node u_i is the father of node v_i.

Next $n + n - 1$ lines describe tree T_b.

The following n lines each contains two integers u_i, v_i, representing that the node with lable i of tree T_b refers to the undirected edge (u_i, v_i) in graph G.

The following $n - 1$ lines each contains two integers u_i, v_i, representing an edge (u_i, v_i) in tree T_b, and the node u_i is the father of node v_i.

All u_i, v_i mentioned above satisfies $1 \leq u_i, v_i \leq m$

Output

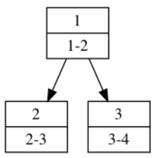
For each test case, print n lines, the i-th line contains an integer refers to the answer of index i.

Example

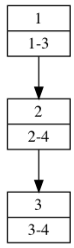
standard input	standard output
2	2
3 4	1
1 2	1
2 3	98
3 4	96
1 2	
1 3	
1 3	
2 4	
3 4	
1 2	
2 3	
2 100	
100 99	
99 98	
1 2	
97 96	
95 94	
1 2	

Explanation

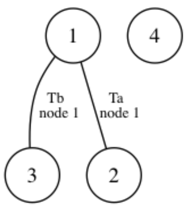
For first test case, the tree T_a and tree T_b look like following photos:



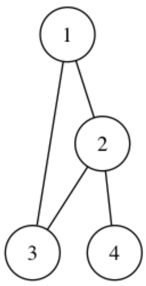
Ta



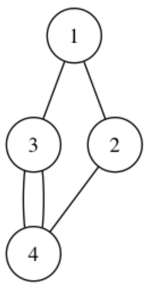
Tb



index 1



index 2



index 3

Problem I. Master of Matrix

There is a matrix of n rows and m columns, with a number a_{ij} in each cell. You can change the number in a cell to its half (round down to the floor, $\lfloor \frac{number}{2} \rfloor$) with cost w_{ij}. For each cell, you can do this operation as many as you need.

You are required to do several operations to make some pairs of specific cells to satisfy the following conditions:

1. Sum of two specific adjacent cells $a_{x_{i1},y_{i1}} + a_{x_{i2},y_{i2}} \leq d_i$
2. Sum of two specific adjacent cells $a_{x_{i1},y_{i1}} + a_{x_{i2},y_{i2}} \geq c_i$

Try to find the minimum cost to satisfy all the conditions, or print "`impossible`" if there is no such solution.

Input

The first line contains an integer T ($1 \leq T \leq 35$) representing the number of test cases.

For each test case, the first line contains two integers n ($1 \leq n \leq 25$) and m ($1 \leq m \leq 25$), representing the number of rows and columns.

The following n lines each contains the numbers a_{ij} ($1 \leq a_{ij} \leq 10000$), each line contains m integers, representing the number in matrix.

The following n lines each contains the costs w_{ij} ($1 \leq w_{ij} \leq 10000$), each line contains m integers representing the cost of changing cells in matrix.

Then, next one following line contains an integer m_1, representing the numbers of first condition need to be satisfied.

The following m_1 ($0 \leq m_1 \leq (n-1)*m + (m-1)*n$) lines, the i-th line contains five integers x_{i1}, y_{i1}, x_{i2}, y_{i2}, d_i, representing a first condition $a_{x_{i1},y_{i1}} + a_{x_{i2},y_{i2}} \leq d_i$ ($1 \leq x_{i1}, x_{i2} \leq n, 1 \leq y_{i1}, y_{i2} \leq m, 0 \leq d_i \leq 20000$).

Then, next one following line contains an integer m_2, representing the numbers of second condition need to be satisfied.

The following m_2 ($0 \leq m_2 \leq (n-1)*m + (m-1)*n$) lines, the i-th line contains five integers x_{i1}, y_{i1}, x_{i2}, y_{i2}, c_i, representing a second condition $a_{x_{i1},y_{i1}} + a_{x_{i2},y_{i2}} \geq c_i$ ($1 \leq x_{i1}, x_{i2} \leq n, 1 \leq y_{i1}, y_{i2} \leq m, 0 \leq c_i \leq 20000$).

It is guaranteed that all (x_{i1}, y_{i1}) and (x_{i2}, y_{i2}) are adjacent.

Output

For each test case, print an integer indicates the minimum cost or "`impossible`" in one line.

Example

standard input	standard output
2 2 2 10 7 5 8 1 2 2 1 4 1 1 1 2 7 1 1 2 1 5 2 2 2 1 6 2 2 1 2 5 4 1 1 1 2 5 1 1 2 1 4 2 2 2 1 6 2 2 1 2 5 2 2 10 7 5 8 1 2 2 1 4 1 1 1 2 5 1 1 2 1 3 2 2 2 1 4 2 2 1 2 6 4 1 1 1 2 1 1 1 2 1 1 2 2 2 1 2 2 2 1 2 4	impossible 9

Explanation

For the first test case, whatever operations we did, there is no solution to meet all conditions given in this test case.

For the second test case, we do three operations in cell $(1, 1)$, one operation in cell $(1, 2)$, one operation in cell $(2, 1)$, two operations in cell $(1, 2)$. Then we meet all conditions and the cost of operations is $3 * 1 + 1 * 2 + 1 * 2 + 2 * 1 = 9$.

Problem J. Master of GCD

Hakase has n numbers in a line. At first, they are all equal to 1. Besides, Hakase is interested in primes. She will choose a continuous subsequence $[l, r]$ and a prime parameter x each time and for every $l \leq i \leq r$, she will change a_i into $a_i * x$. To simplify the problem, x will be 2 or 3. After m operations, Hakase wants to know what is the greatest common divisor of all the numbers.

Input

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

For each test case, the first line contains two integers n $(1 \leq n \leq 100000)$ and m $(1 \leq m \leq 100000)$, where n refers to the length of the whole sequence and m means there are m operations.

The following m lines, each line contains three integers l_i $(1 \leq l_i \leq n)$, r_i $(1 \leq r_i \leq n)$, x_i $(x_i \in \{2, 3\})$, which are referred above.

Output

For each test case, print an integer in one line, representing the greatest common divisor of the sequence. **Due to the answer might be very large, print the answer modulo 998244353.**

Example

<table>
<tr><th>standard input</th><th>standard output</th></tr>
<tr><td><pre>2
5 3
1 3 2
3 5 2
1 5 3
6 3
1 2 2
5 6 2
1 6 2</pre></td><td><pre>6
2</pre></td></tr>
</table>

Explanation

For the first test case, after all operations, the numbers will be $[6, 6, 12, 6, 6]$. So the greatest common divisor is 6.

Problem K. Master of Sequence

There are two sequences $a_1, a_2, \cdots, a_n$, $b_1, b_2, \cdots, b_n$. Let $S(t) = \sum_{i=1}^{n} \lfloor \frac{t-b_i}{a_i} \rfloor$. There are m operations within three kinds as following:

- `1 x y`: change value a_x to y.
- `2 x y`: change value b_x to y.
- `3 k`: ask $\min\{t | k \le S(t)\}$

Input

The first line contains a integer T $(1 \le T \le 5)$ representing the number of test cases.

For each test case, the first line contains two integers n $(1 \le n \le 100000)$, m $(1 \le m \le 10000)$.

The following line contains n integers representing the sequence $a_1, a_2, \cdots, a_n$.

The following line contains n integers representing the sequence $b_1, b_2, \cdots, b_n$.

The following m lines, each line contains two or three integers representing an operation mentioned above.

It is guaranteed that, at any time, we have $1 \le a_i \le 1000$, $1 \le b_i, k \le 10^9$. And the number of queries (type 3 operation) in each test case will not exceed 1000.

Output

For each query operation (type 3 operation), print the answer in one line.

Example

standard input	standard output
2 4 6 2 4 6 8 1 3 5 7 1 2 3 2 3 3 3 15 1 3 8 3 90 3 66 8 5 2 4 8 3 1 3 6 24 2 2 39 28 85 25 98 35 3 67 3 28 3 73 3 724 3 7775	17 87 65 72 58 74 310 2875

Problem L. Mod, Xor and Everything

You are given an integer n.

You are required to calculate `(n mod 1) xor (n mod 2) xor ... xor (n mod (n - 1)) xor (n mod n)`.

The "`xor`" operation means "exclusive OR".

Input

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

For each test case, there is an integer n $(1 \leq n \leq 10^{12})$ in one line.

Output

For each test case, print the answer in one line.

Example

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