Codeforces Round #680 (Div. 1, based on Moscow Team Olympiad)

A. Division

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

Oleg's favorite subjects are History and Math, and his favorite branch of mathematics is division.

To improve his division skills, Oleg came up with t pairs of integers p_i and q_i and for each pair decided to find the **greatest** integer x_i, such that:

- p_i is divisible by x_i;
- x_i is not divisible by q_i.

Oleg is really good at division and managed to find all the answers quickly, how about you?

Input

The first line contains an integer t ($1 \le t \le 50$) — the number of pairs.

Each of the following t lines contains two integers p_i and q_i ($1 \le p_i \le 10^{18}$; $2 \le q_i \le 10^9$) — the i-th pair of integers.

Output

Print t integers: the i-th integer is the largest x_i such that p_i is divisible by x_i, but x_i is not divisible by q_i.

One can show that there is always at least one value of x_i satisfying the divisibility conditions for the given constraints.

Example

input
3 10 4 12 6 179 822
output
10 4 179

Note

For the first pair, where $p_1 = 10$ and $q_1 = 4$, the answer is $x_1 = 10$, since it is the greatest divisor of 10 and 10 is not divisible by 4.

For the second pair, where $p_2 = 12$ and $q_2 = 6$, note that

- 12 is not a valid x_2, since 12 is divisible by $q_2 = 6$;
- 6 is not valid x_2 as well: 6 is also divisible by $q_2 = 6$.

The next available divisor of $p_2 = 12$ is 4, which is the answer, since 4 is not divisible by 6.

B. Divide and Sum

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given an array a of length $2n$. Consider a partition of array a into two subsequences p and q of length n each (each element of array a should be in exactly one subsequence: either in p or in q).

Let's sort p in non-decreasing order, and q in non-increasing order, we can denote the sorted versions by x and y, respectively. Then the *cost* of a partition is defined as $f(p, q) = \sum_{i=1}^{n} |x_i - y_i|$.

Find the sum of $f(p, q)$ over all correct partitions of array a. Since the answer might be too big, print its remainder modulo 998244353.

Input

The first line contains a single integer n ($1 \le n \le 150\,000$).

The second line contains $2n$ integers $a_1, a_2, \ldots, a_{2n}$ ($1 \le a_i \le 10^9$) — elements of array a.

Output
Print one integer — the answer to the problem, modulo 998244353.

Examples

input
1 1 4
output
6

input
2 2 1 2 1
output
12

input
3 2 2 2 2 2 2
output
0

input
5 13 8 35 94 9284 34 54 69 123 846
output
2588544

Note
Two partitions of an array are considered different if the sets of indices of elements included in the subsequence p are different.

In the first example, there are two correct partitions of the array a:

1. $p = [1]$, $q = [4]$, then $x = [1]$, $y = [4]$, $f(p, q) = |1 - 4| = 3$;
2. $p = [4]$, $q = [1]$, then $x = [4]$, $y = [1]$, $f(p, q) = |4 - 1| = 3$.

In the second example, there are six valid partitions of the array a:

1. $p = [2, 1]$, $q = [2, 1]$ (elements with indices 1 and 2 in the original array are selected in the subsequence p);
2. $p = [2, 2]$, $q = [1, 1]$;
3. $p = [2, 1]$, $q = [1, 2]$ (elements with indices 1 and 4 are selected in the subsequence p);
4. $p = [1, 2]$, $q = [2, 1]$;
5. $p = [1, 1]$, $q = [2, 2]$;
6. $p = [2, 1]$, $q = [2, 1]$ (elements with indices 3 and 4 are selected in the subsequence p).

C. Team-Building

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

The new academic year has started, and Berland's university has n first-year students. They are divided into k academic groups, however, some of the groups might be empty. Among the students, there are m pairs of acquaintances, and each acquaintance pair might be both in a common group or be in two different groups.

Alice is the curator of the first years, she wants to host an entertaining game to make everyone know each other. To do that, she will select two different academic groups and then divide the students of those groups into two teams. The game requires that there are no acquaintance pairs inside each of the teams.

Alice wonders how many pairs of groups she can select, such that it'll be possible to play a game after that. All students of the two selected groups must take part in the game.

Please note, that the teams Alice will form for the game don't need to coincide with groups the students learn in. Moreover, teams may have different sizes (or even be empty).

Input
The first line contains three integers n, m and k ($1 \le n \le 500\,000$; $0 \le m \le 500\,000$; $2 \le k \le 500\,000$) — the number of students, the number of pairs of acquaintances and the number of groups respectively.

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le k$), where c_i equals to the group number of the i-th student.

Next m lines follow. The i-th of them contains two integers a_i and b_i ($1 \leq a_i, b_i \leq n$), denoting that students a_i and b_i are acquaintances. It's guaranteed, that $a_i \neq b_i$, and that no (unordered) pair is mentioned more than once.

Output

Print a single integer — the number of ways to choose two different groups such that it's possible to select two teams to play the game.

Examples

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

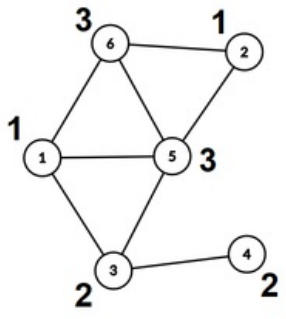
input
4 3 3 1 1 2 2 1 2 2 3 3 4
output
3

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

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

Note

The acquaintances graph for the first example is shown in the picture below (next to each student there is their group number written).



In that test we can select the following groups:

- Select the first and the second groups. For instance, one team can be formed from students 1 and 4, while other team can be formed from students 2 and 3.
- Select the second and the third group. For instance, one team can be formed 3 and 6, while other team can be formed from

students 4 and 5.
- We can't select the first and the third group, because there is no way to form the teams for the game.

In the second example, we can select any group pair. Please note, that even though the third group has no students, we still can select it (with some other group) for the game.

D. Rectangular Polyline

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

One drew a closed polyline on a plane, that consisted only of vertical and horizontal segments (parallel to the coordinate axes). The segments alternated between horizontal and vertical ones (a horizontal segment was always followed by a vertical one, and vice versa). The polyline did not contain strict self-intersections, which means that in case any two segments shared a common point, that point was an endpoint for both of them (please consult the examples in the notes section).

Unfortunately, the polyline was erased, and you only know the lengths of the horizonal and vertical segments. Please construct any polyline matching the description with such segments, or determine that it does not exist.

Input
The first line contains one integer t ($1 \le t \le 200$) —the number of test cases.

The first line of each test case contains one integer h ($1 \le h \le 1000$) — the number of horizontal segments. The following line contains h integers $l_1, l_2, \ldots, l_h$ ($1 \le l_i \le 1000$) — lengths of the horizontal segments of the polyline, in arbitrary order.

The following line contains an integer v ($1 \le v \le 1000$) — the number of vertical segments, which is followed by a line containing v integers $p_1, p_2, \ldots, p_v$ ($1 \le p_i \le 1000$) — lengths of the vertical segments of the polyline, in arbitrary order.

Test cases are separated by a blank line, and the sum of values $h + v$ over all test cases does not exceed 1000.

Output
For each test case output Yes, if there exists at least one polyline satisfying the requirements, or No otherwise. If it does exist, in the following n lines print the coordinates of the polyline vertices, in order of the polyline traversal: the i-th line should contain two integers x_i and y_i — coordinates of the i-th vertex.

Note that, each polyline segment must be either horizontal or vertical, and the segments should alternate between horizontal and vertical. The coordinates should not exceed 10^9 by their absolute value.

Examples

input
2 2 1 1 2 1 1 2 1 2 2 3 3
output
Yes 1 0 1 1 0 1 0 0 No

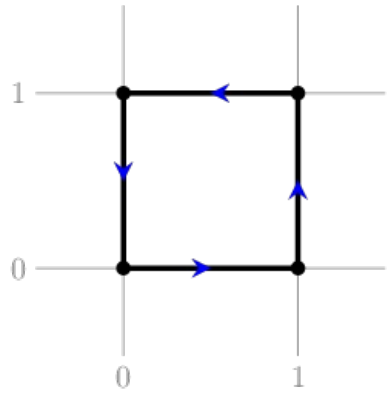
input
2 4 1 1 1 1 4 1 1 1 1 3 2 1 1 3 2 1 1
output
Yes 1 0 1 1 2 1 2 2 1 2 1 1 0 1

```
0 0
Yes
0 -2
2 -2
2 -1
1 -1
1 0
0 0
```

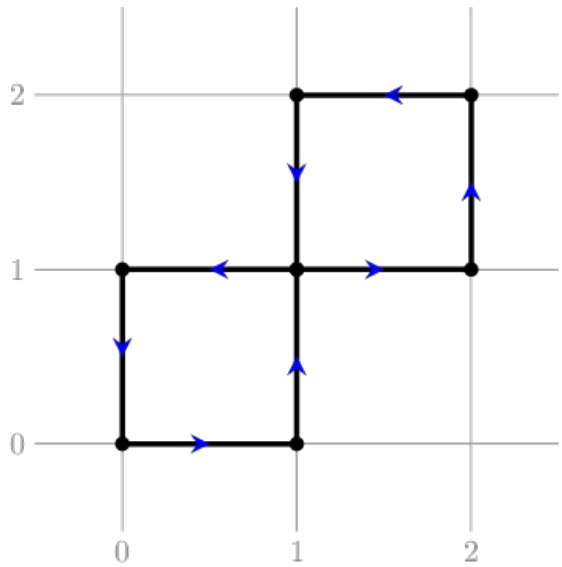
input
2 4 1 4 1 2 4 3 4 5 12 4 1 2 3 6 2 1 3
output
Yes 2 0 2 3 3 3 3 7 4 7 4 12 0 12 0 0 No

Note

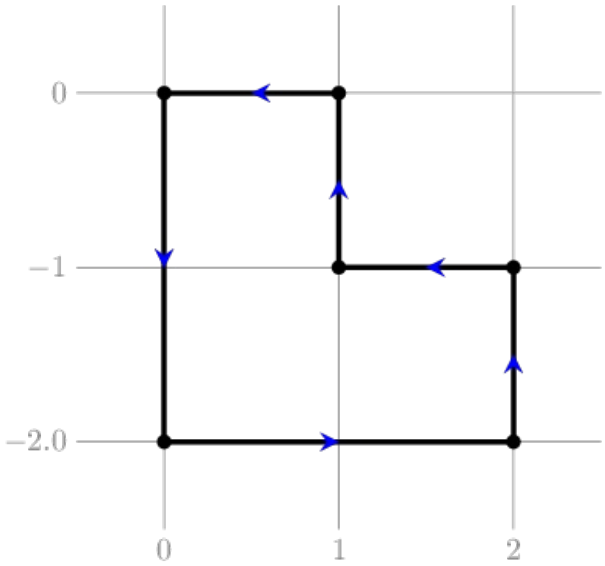
In the first test case of the first example, the answer is Yes — for example, the following picture illustrates a square that satisfies the requirements:



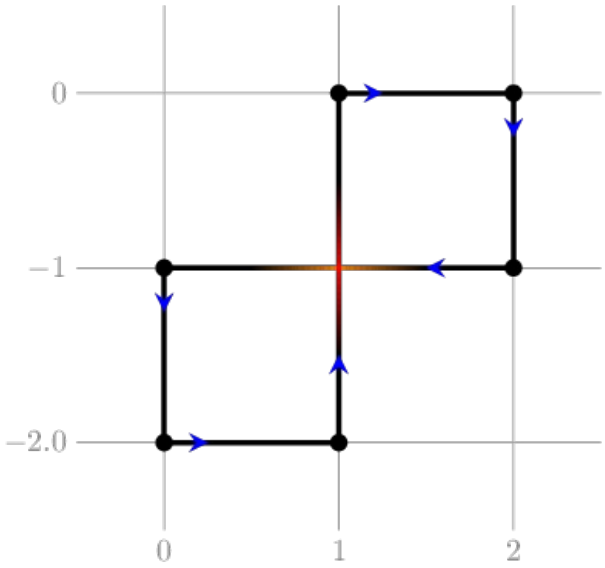
In the first test case of the second example, the desired polyline also exists. Note that, the polyline contains self-intersections, but only in the endpoints:



In the second test case of the second example, the desired polyline could be like the one below:



Note that the following polyline is **not** a valid one, since it contains self-intersections that are not endpoints for some of the segments:



E. Finding the Vertex

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

This is an interactive problem.

You are given a tree — connected undirected graph without cycles. One vertex of the tree is special, and you have to find which one. You can ask questions in the following form: given an edge of the tree, which endpoint is closer to the special vertex, meaning which endpoint's shortest path to the special vertex contains fewer edges. You have to find the special vertex by asking the minimum number of questions in the worst case for a given tree.

Please note that the special vertex might not be fixed by the interactor in advance: it might change the vertex to any other one, with the requirement of being consistent with the previously given answers.

Input

You are given an integer n ($2 \le n \le 100$) — the number of vertices in a tree.

The folloiwing $n-1$ lines contain two integers each, u and v ($1 \le u, v \le n$), that denote an edge in the tree connecting u and v. It is guaranteed that the given edges form a tree.

Interaction

After reading the input data, one can start making queries. There are two possible queries:

1. "? u v" — to ask for an edge (u, v) ($1 \le u, v \le n$) which of the endpoints is closer to the special vertex. The answer to this query is one of the endpoints. Note that, u and v must be connected by an edge, and hence they can not have the same distance to the special vertex.
2. "! u" — to indicate that you found the special vertex. After the program does that, it must immediately terminate.

Do not forget to output the end of line and flush the output. Otherwise you will get `Idleness limit exceeded` verdict. To flush the output, you can use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `sys.stdout.flush()` in Python;
- see documentation for other languages.

In case you ask more queries than needed in the worst case for a given tree, you will get verdict `Wrong answer`.

Examples

input
5 1 2 2 3 3 4 4 5 3 2 1
output
? 3 4 ? 2 3 ? 1 2 ! 1

input
5 2 1 3 1 4 1 5 1 1 1 4
output
? 1 2 ? 1 3 ? 1 4 ! 4

Note

Hacks are forbidden in this task.