A. Recovering a Small String

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

Nikita had a word consisting of exactly 3 lowercase Latin letters. The letters in the Latin alphabet are numbered from 1 to 26, where the letter "a" has the index 1, and the letter "z" has the index 26.

He encoded this word as the sum of the positions of all the characters in the alphabet. For example, the word "cat" he would encode as the integer $3 + 1 + 20 = 24$, because the letter "c" has the index 3 in the alphabet, the letter "a" has the index 1, and the letter "t" has the index 20.

However, this encoding turned out to be ambiguous! For example, when encoding the word "ava", the integer $1 + 22 + 1 = 24$ is also obtained.

Determine the lexicographically smallest word of 3 letters that could have been encoded.

A string a is lexicographically smaller than a string b if and only if one of the following holds:

- a is a prefix of b, but $a \neq b$;
- in the first position where a and b differ, the string a has a letter that appears earlier in the alphabet than the corresponding letter in b.

Input

The first line of the input contains a single integer t ($1 \leq t \leq 100$) — the number of test cases in the test.

This is followed by the descriptions of the test cases.

The first and only line of each test case contains an integer n ($3 \leq n \leq 78$) — the encoded word.

Output

For each test case, output the lexicographically smallest three-letter word that could have been encoded on a separate line.

Standard Input	Standard Output
5 24 70 3 55 48	aav rzz aaa czz auz

,

B. Make Equal

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

There are n containers of water lined up, numbered from left to right from 1 to n. Each container can hold any amount of water; initially, the i-th container contains a_i units of water. The sum of a_i is divisible by n.

You can apply the following operation any (possibly zero) number of times: pour any amount of water from the i-th container to the j-th container, where i must be **less** than j (i.e. $i < j$). Any index can be chosen as i or j any number of times.

Determine whether it is possible to make the amount of water in all containers the same using this operation.

Input

The first line of the input contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. Then the descriptions of the test cases follow.

The first line of each test case contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the number of containers with water.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$) — the amounts of water in the containers. It is guaranteed that the sum of a_i in each test case does not exceed $2 \cdot 10^9$. Also, the sum of a_i is divisible by n.

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

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 make the amount of water in all containers the same using the described operation. Otherwise, output "NO".

You can output each letter in any case (lowercase or uppercase). For example, the strings "yEs", "yes", "Yes", and "YES" will be accepted as a positive answer.

Standard Input	Standard Output
6 1 43 2 1 3 5 4 5 2 1 3 3 1 2 3 7 4 5 5 0 6 4 4 7 6 5 5 1 3 4 4	YES NO YES NO NO YES

Note

In the third test case of the example ($a = [4, 5, 2, 1, 3]$), you can proceed as follows:

- pour 1 unit of water from the first vessel to the fourth, then $a = [3, 5, 2, 2, 3]$;
- pour 1 unit of water from the second vessel to the third, then $a = [3, 4, 3, 2, 3]$;
- pour 1 unit of water from the second vessel to the fourth, then $a = [3, 3, 3, 3, 3]$.

,

C. Make Equal Again

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

You have an array a of n integers.

You can **no more than once** apply the following operation: select three integers i, j, x ($1 \le i \le j \le n$) and assign all elements of the array with indexes from i to j the value x. The price of this operation depends on the selected indices and is equal to $(j - i + 1)$ burles.

For example, the array is equal to $[1, 2, 3, 4, 5, 1]$. If we choose $i = 2, j = 4, x = 8$, then after applying this operation, the array will be equal to $[1, 8, 8, 8, 5, 1]$.

What is the least amount of burles you need to spend to make all the elements of the array equal?

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of input test cases. The descriptions of the test cases follow.

The first line of the description of each test case contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the size of the array.

The second line of the description of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$) — array elements.

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

Output

For each test case, output one integer — the minimum number of burles that will have to be spent to make all the elements of the array equal. It can be shown that this can always be done.

Standard Input	Standard Output
8 6 1 2 3 4 5 1 7 1 1 1 1 1 1 1 8 8 8 8 1 2 8 8 8 1 1 2 1 2 3 1 2 3 7 4 3 2 7 1 1 3 9	4 0 2 0 1 2 6 7

9 9 2 9 2 5 5 5 3	

,

D. Divisible Pairs

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

Polycarp has two favorite integers x and y (they can be equal), and he has found an array a of length n.

Polycarp considers a pair of indices $\langle i, j \rangle$ ($1 \le i < j \le n$) *beautiful* if:

- $a_i + a_j$ is divisible by x;
- $a_i - a_j$ is divisible by y.

For example, if $x = 5$, $y = 2$, $n = 6$, $a =$[$1, 2, 7, 4, 9, 6$], then the only *beautiful* pairs are:

- $\langle 1, 5 \rangle$: $a_1 + a_5 = 1 + 9 = 10$ (10 is divisible by 5) and $a_1 - a_5 = 1 - 9 = -8$ (-8 is divisible by 2);
- $\langle 4, 6 \rangle$: $a_4 + a_6 = 4 + 6 = 10$ (10 is divisible by 5) and $a_4 - a_6 = 4 - 6 = -2$ (-2 is divisible by 2).

Find the number of *beautiful* pairs in the array a.

Input

The first line of the input contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. Then the descriptions of the test cases follow.

The first line of each test case contains three integers n, x, and y ($2 \le n \le 2 \cdot 10^5$, $1 \le x, y \le 10^9$) — the size of the array and Polycarp's favorite integers.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the elements of the array.

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 a single integer — the number of *beautiful* pairs in the array a.

Standard Input	Standard Output
7 6 5 2 1 2 7 4 9 6 7 9 5 1 10 15 3 8 12 15 9 4 10 14 10 2 2 11 11 13 5 6 9 5 6 10 7 6 7 9 7 7 10 10 9 6 2 4 9 7 1 2 2 13 3 15 9 2 3 14 6 1 15 12 15 8 2 15	2 0 1 3 5 7 0

10 5 7 13 3 3 2 12 11 3 7 13 14	

,

E. Anna and the Valentine's Day Gift

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

Sasha gave Anna a list a of n integers for Valentine's Day. Anna doesn't need this list, so she suggests destroying it by playing a game.

Players take turns. Sasha is a gentleman, so he gives Anna the right to make the first move.

- On her turn, **Anna must** choose an element a_i from the list and *reverse* the sequence of its digits. For example, if Anna chose the element with a value of 42, it would become 24; if Anna chose the element with a value of 1580, it would become 851. Note that leading zeros are removed. After such a turn, the number of elements in the list does not change.
- On his turn, **Sasha must** extract **two** elements a_i and a_j ($i \neq j$) from the list, *concatenate* them in any order and insert the result back into the list. For example, if Sasha chose the elements equal to 2007 and 19, he would remove these two elements from the list and add the integer 200719 or 192007. After such a turn, the number of elements in the list decreases by 1.

Players can't skip turns. The game ends when Sasha can't make a move, i.e. **after** Anna's move there is **exactly** one number left in the list. If this integer is **not less than** 10^m (i.e., $\geq 10^m$), Sasha wins. Otherwise, Anna wins.

It can be shown that the game will always end. Determine who will win if both players play optimally.

Input

The first line contains an integer t ($1 \leq t \leq 10^4$) — the number of test cases.

Then follows the description of the test cases.

The first line of each test case contains integers n, m ($1 \leq n \leq 2 \cdot 10^5$, $0 \leq m \leq 2 \cdot 10^6$) — the number of integers in the list and the parameter determining when Sasha wins.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^9$) — the list that Sasha gave to Anna.

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

Output

For each test case, output:

- "Sasha", if Sasha wins with optimal play;
- "Anna", if Anna wins with optimal play.

Standard Input	Standard Output
9 2 2 14 2 3 5 9 56 1	Sasha Anna Anna Sasha Sasha

4 10 1 2007 800 1580 4 5 5000 123 30 4 10 10 6 4 6 2 3 1 10 9 10 7 1 1 6 1 1 10 8 9 1 2 9 10 10 2 10 2 4 5 10 10 10 10	Anna Anna Anna Sasha

Note

Consider the first test case.

Anna can reverse the integer 2, then Sasha can concatenate the integers 2 and 14, obtaining the integer 214, which is greater than $10^2 = 100$. If Anna had reversed the integer 14, Sasha would have concatenated the integers 41 and 2, obtaining the integer 412, which is greater than $10^2 = 100$. Anna has no other possible moves, so she loses.

,

F. Chat Screenshots

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

There are n people in the programming contest chat. Chat participants are ordered by activity, but each person sees himself at the top of the list.

For example, there are 4 participants in the chat, and their order is $[2, 3, 1, 4]$. Then

- 1-st user sees the order $[1, 2, 3, 4]$.
- 2-nd user sees the order $[2, 3, 1, 4]$.
- 3-rd user sees the order $[3, 2, 1, 4]$.
- 4-th user sees the order $[4, 2, 3, 1]$.

k people posted screenshots in the chat, which show the order of participants shown to this user. The screenshots were taken within a short period of time, and the order of participants has not changed.

Your task is to determine whether there is a certain order that all screenshots correspond to.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of input test cases. The descriptions of test cases follow.

The first line of the description of each test case contains two integers n and k ($1 \le k \le n \le 2 \cdot 10^5, n \cdot k \le 2 \cdot 10^5$) — the number of chat participants and the number of participants who posted screenshots.

The following k lines contain descriptions of screenshots posted by the participants.

The i-th row contains n integers a_{ij} each ($1 \le a_{ij} \le n$, all a_{ij} are different) — the order of participants shown to the participant a_{i0}, where a_{i0} — the author of the screenshot. You can show that in the screenshot description it will always be at the top of the list.

It is guaranteed that the sum of $n \cdot k$ for all test cases does not exceed $2 \cdot 10^5$. It is also guaranteed that all the authors of the screenshots are different.

Output

Output t lines, each of which is the answer to the corresponding test case. As an answer, output "YES" if there exists at least one order of participants, under which all k screenshots could have been obtained. Otherwise, output "NO".

You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

Standard Input	Standard Output
10 5 1 1 2 3 4 5 4 4	YES YES YES YES

1 2 3 4 2 3 1 4 3 2 1 4 4 2 3 1 6 2 1 3 5 2 4 6 6 3 5 2 1 4 3 3 1 2 3 2 3 1 3 2 1 10 2 1 2 3 4 5 6 7 8 9 10 10 9 8 7 6 5 4 3 2 1 1 1 1 5 2 1 2 3 5 4 2 1 3 5 4 3 3 3 1 2 2 3 1 1 3 2 5 4 3 5 1 4 2 2 5 1 4 3 1 5 4 3 2 5 1 4 3 2 3 3 1 3 2 2 1 3 3 2 1	NO YES YES YES YES NO

,

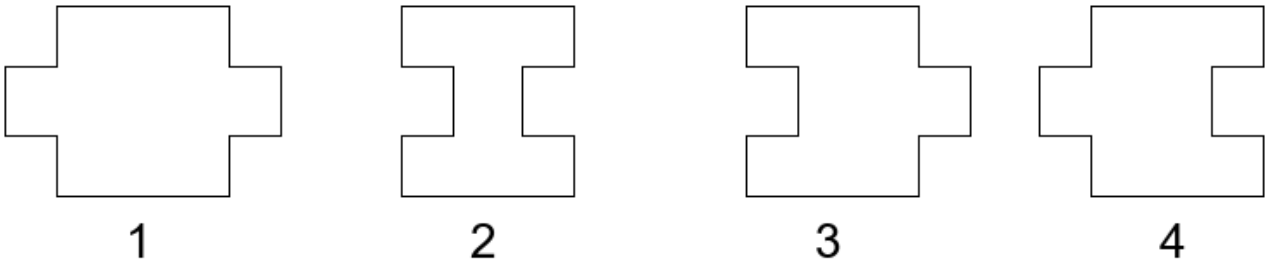
G. One-Dimensional Puzzle

```
Input file:     standard input
Output file:    standard output
Time limit:     4 seconds
Memory limit:   256 megabytes
```

You have a one-dimensional puzzle, all the elements of which need to be put in one row, connecting with each other. All the puzzle elements are completely white and distinguishable from each other only if they have different shapes.

Each element has straight borders at the top and bottom, and on the left and right it has connections, each of which can be a protrusion or a recess. You **cannot** rotate the elements.

You can see that there are exactly 4 types of elements. Two elements can be connected if the right connection of the left element is opposite to the left connection of the right element.



All possible types of elements.

The puzzle contains c_1, c_2, c_3, c_4 elements of each type. The puzzle is considered complete if you have managed to combine **all** elements into one long chain. You want to know how many ways this can be done.

Input

The first line contains a single integer t ($1 \le t \le 2 \cdot 10^5$) — the number of input test cases. The descriptions of the test cases follow.

The description of each test case contains 4 integers c_i ($0 \le c_i \le 10^6$) — the number of elements of each type, respectively.

It is guaranteed that the sum of c_i for all test cases does not exceed $4 \cdot 10^6$.

Output

For each test case, print one integer — the number of possible ways to solve the puzzle.

Two methods are considered different if there is i, such that the types of elements at the i position in these methods differ.

Since the answer can be very large, output it modulo 998244353.

If it is impossible to solve the puzzle, print 0.

Standard Input	Standard Output
11 1 1 1 1 1 2 5 10 4 6 100 200	4 66 0 794100779

900000 900000 900000 900000 0 0 0 0 0 0 566 239 1 0 0 0 100 0 100 0 0 0 0 4 5 5 0 2 5 4 0 5	1 0 1 0 1 36 126