Educational Codeforces Round 159 (Rated for Div. 2)

A. Binary Imbalance

2 seconds, 256 megabytes

You are given a string s, consisting only of characters '0' and/or '1'.

In one operation, you choose a position i from 1 to $|s| - 1$, where $|s|$ is the current length of string s. Then you insert a character between the i-th and the $(i + 1)$-st characters of s. If $s_i = s_{i+1}$, you insert '1'. If $s_i \neq s_{i+1}$, you insert '0'.

Is it possible to make the number of zeroes in the string strictly greater than the number of ones, using any number of operations (possibly, none)?

Input

The first line contains a single integer t ($1 \le t \le 100$) — the number of testcases.

The first line of each testcase contains an integer n ($1 \le n \le 100$).

The second line contains a string s of length exactly n, consisting only of characters '0' and/or '1'.

Output

For each testcase, print "YES" if it's possible to make the number of zeroes in s strictly greater than the number of ones, using any number of operations (possibly, none). Otherwise, print "NO".

input
3 2 00 2 11 2 10

output
YES NO YES

In the first testcase, the number of zeroes is already greater than the number of ones.

In the second testcase, it's impossible to insert any zeroes in the string.

In the third testcase, you can choose $i = 1$ to insert a zero between the 1-st and the 2-nd characters. Since $s_1 \neq s_2$, you insert a '0'. The resulting string is "100". It has two zeroes and only a single one, so the answer is "YES".

B. Getting Points

1 second, 256 megabytes

Monocarp is a student at Berland State University. Due to recent changes in the Berland education system, Monocarp has to study only one subject — programming.

The academic term consists of n days, and in order not to get expelled, Monocarp has to earn at least P points during those n days. There are two ways to earn points — completing practical tasks and attending lessons. For each practical task Monocarp fulfills, he earns t points, and for each lesson he attends, he earns l points.

Practical tasks are unlocked "each week" as the term goes on: the first task is unlocked on day 1 (and can be completed on any day from 1 to n), the second task is unlocked on day 8 (and can be completed on any day from 8 to n), the third task is unlocked on day 15, and so on.

Every day from 1 to n, there is a lesson which can be attended by Monocarp. And every day, Monocarp chooses whether to study or to rest the whole day. When Monocarp decides to study, he attends a lesson and can complete **no more than** 2 tasks, which are already unlocked and not completed yet. If Monocarp rests the whole day, he skips a lesson and ignores tasks.

Monocarp wants to have as many days off as possible, i. e. he wants to maximize the number of days he rests. Help him calculate the maximum number of days he can rest!

Input

The first line contains a single integer tc ($1 \le tc \le 10^4$) — the number of test cases. The description of the test cases follows.

The only line of each test case contains four integers n, P, l and t ($1 \le n, l, t \le 10^9$; $1 \le P \le 10^{18}$) — the number of days, the minimum total points Monocarp has to earn, the points for attending one lesson and points for completing one task.

It's guaranteed for each test case that it's possible not to be expelled if Monocarp will attend all lessons and will complete all tasks.

Output

For each test, print one integer — the maximum number of days Monocarp can rest without being expelled from University.

input
5 1 5 5 2 14 3000000000 1000000000 500000000 100 20 1 10 8 120 10 20 42 280 13 37

output
0 12 99 0 37

In the first test case, the term lasts for 1 day, so Monocarp should attend at day 1. Since attending one lesson already gives 5 points ($5 \ge P$), so it doesn't matter, will Monocarp complete the task or not.

In the second test case, Monocarp can, for example, study at days 8 and 9: at day 8 he will attend a lesson for 10^9 points and complete two tasks for another $5 \cdot 10^8 + 5 \cdot 10^8$ points. And at day 9 he only attends a lesson for another 10^9 points.

In the third test case, Monocarp can, for example, study at day 42: attending a lesson gives him 1 point and solving 2 out of 6 available tasks gives him another $2 \cdot 10$ points.

In the fourth test case, Monocarp has to attend all lessons and complete all tasks to get $8 \cdot 10 + 2 \cdot 20 = 120$ points.

In the fifth test case, Monocarp can, for example, study at days: 8 — one lesson and first and second tasks; 15 — one lesson and the third task; 22 — one lesson and the fourth task; 29 — one lesson and the fifth task; 36 — one lesson and the sixth task.

C. Insert and Equalize

2 seconds, 256 megabytes

You are given an integer array $a_1, a_2, \ldots, a_n$, all its elements are distinct.

First, you are asked to insert one more integer a_{n+1} into this array. a_{n+1} should not be equal to any of $a_1, a_2, \ldots, a_n$.

Then, you will have to make all elements of the array equal. At the start, you choose a **positive** integer x ($x > 0$). In one operation, you add x to exactly one element of the array. **Note that x is the same for all operations**.

What's the smallest number of operations it can take you to make all elements equal, after you choose a_{n+1} and x?

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of testcases.

The first line of each testcase contains a single integer n ($1 \le n \le 2 \cdot 10^5$).

The second line contains n integers $a_1, a_2, \dots, a_n$ ($-10^9 \le a_i \le 10^9$). All a_i are distinct.

The sum of n over all testcases doesn't exceed $2 \cdot 10^5$.

Output

For each testcase, print a single integer — the smallest number of operations it can take you to make all elements equal, after you choose integers a_{n+1} and x.

input
3 3 1 2 3 5 1 -19 17 -3 -15 1 10

output
6 27 1

In the first testcase, you can choose $a_{n+1} = 4$, the array becomes $[1, 2, 3, 4]$. Then choose $x = 1$ and apply the operation 3 times to the first element, 2 times to the second element, 1 time to the third element and 0 times to the fourth element.

In the second testcase, you can choose $a_{n+1} = 13, x = 4$.

In the third testcase, you can choose $a_{n+1} = 9, x = 1$. Then apply the operation once to a_{n+1}.

D. Robot Queries

2 seconds, 256 megabytes

There is an infinite 2-dimensional grid. Initially, a robot stands in the point $(0, 0)$. The robot can execute four commands:

- `U` — move from point (x, y) to $(x, y + 1)$;
- `D` — move from point (x, y) to $(x, y - 1)$;
- `L` — move from point (x, y) to $(x - 1, y)$;
- `R` — move from point (x, y) to $(x + 1, y)$.

You are given a sequence of commands s of length n. Your task is to answer q **independent** queries: given four integers x, y, l and r; determine whether the robot visits the point (x, y), while executing a sequence s, but the substring from l to r is reversed (i.e. the robot performs commands in order $s_1 s_2 s_3 \dots s_{l-1} s_r s_{r-1} s_{r-2} \dots s_l s_{r+1} s_{r+2} \dots s_n$).

Input

The first line contains two integers n and q ($1 \le n, q \le 2 \cdot 10^5$) — the length of the command sequence and the number of queries, respectively.

The second line contains a string s of length n, consisting of characters `U`, `D`, `L` and/or `R`.

Then q lines follow, the i-th of them contains four integers x_i, y_i, l_i and r_i ($-n \le x_i, y_i \le n$; $1 \le l \le r \le n$) describing the i-th query.

Output

For each query, print `YES` if the robot visits the point (x, y), while executing a sequence s, but the substring from l to r is reversed; otherwise print `NO`.

input
8 3 RDLLUURU -1 2 1 7 0 0 3 4 0 1 7 8

output
YES YES NO

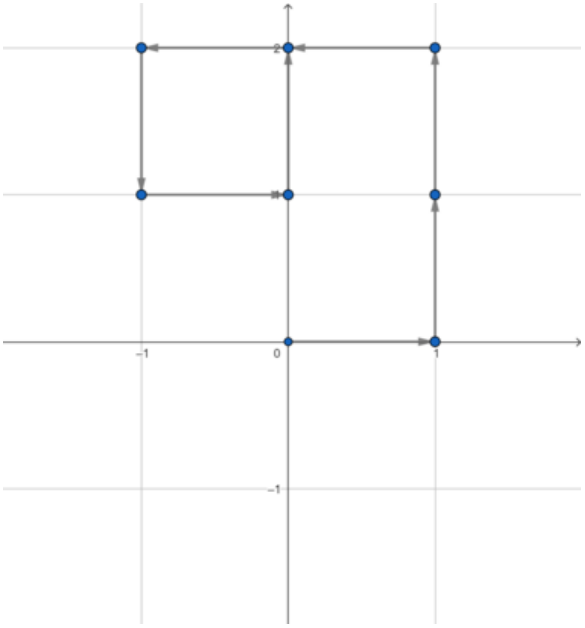
input
4 2 RLDU 0 0 2 2 -1 -1 2 3

output
YES NO

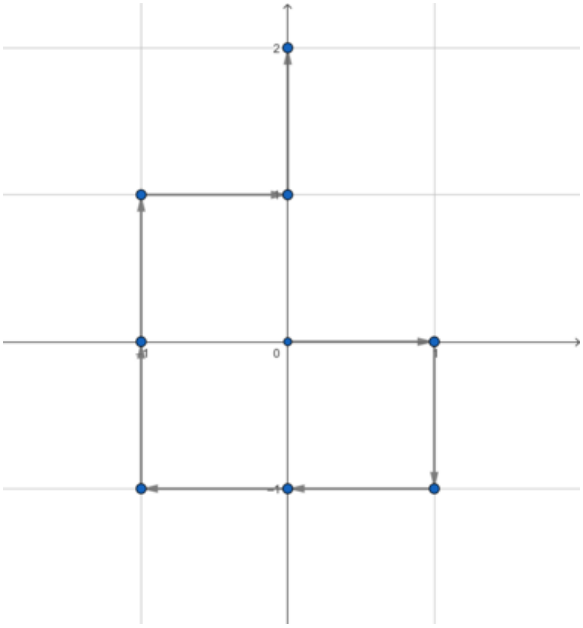
input
10 6 DLUDLRULLD -1 0 1 10 -1 -2 2 5 -4 -2 6 10 -1 0 3 9 0 1 4 7 -3 -1 5 8

output
YES YES YES NO YES YES

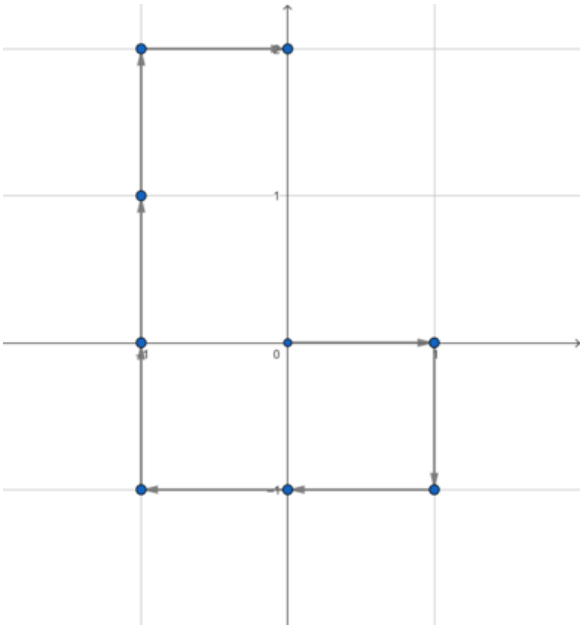
In the first query of the first sample, the path of the robot looks as follows:



In the second query of the first sample, the path of the robot looks as follows:



In the third query of the first sample, the path of the robot looks as follows:



E. Collapsing Strings

2 seconds, 256 megabytes

You are given n strings $s_1, s_2, \ldots, s_n$, consisting of lowercase Latin letters. Let $|x|$ be the length of string x.

Let a *collapse* $C(a, b)$ of two strings a and b be the following operation:

- if a is empty, $C(a, b) = b$;
- if b is empty, $C(a, b) = a$;
- if the last letter of a is equal to the first letter of b, then $C(a, b) = C(a_{1,|a|-1}, b_{2,|b|})$, where $s_{l,r}$ is the substring of s from the l-th letter to the r-th one;
- otherwise, $C(a, b) = a + b$, i.e. the concatenation of two strings.

Calculate $\sum_{i=1}^{n} \sum_{j=1}^{n} |C(s_i, s_j)|$.

Input

The first line contains a single integer n ($1 \le n \le 10^6$).

Each of the next n lines contains a string s_i ($1 \le |s_i| \le 10^6$), consisting of lowercase Latin letters.

The total length of the strings doesn't exceed 10^6.

Output

Print a single integer — $\sum_{i=1}^{n} \sum_{j=1}^{n} |C(s_i, s_j)|$.

input
3 aba ab ba
output
20

input
5 abab babx xab xba bab
output
126

F. Trees and XOR Queries Again

6.5 seconds, 512 megabytes

You are given a tree consisting of n vertices. There is an integer written on each vertex; the i-th vertex has integer a_i written on it.

You have to process q queries. The i-th query consists of three integers x_i, y_i and k_i. For this query, you have to answer if it is possible to choose a set of vertices $v_1, v_2, \ldots, v_m$ (possibly empty) such that:

- every vertex v_j is on the simple path between x_i and y_i (endpoints can be used as well);
- $a_{v_1} \oplus a_{v_2} \oplus \cdots \oplus a_{v_m} = k_i$, where $\oplus$ denotes the bitwise XOR operator.

Input

The first line contains one integer n ($2 \le n \le 2 \cdot 10^5$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 2^{20} - 1$).

Then $n - 1$ lines follow. Each of them contains two integers u and v ($1 \le u, v \le n$; $u \ne v$) denoting an edge of the tree.

The next line contains one integer q ($1 \le q \le 2 \cdot 10^5$) — the number of queries.

Then q lines follow. The i-th of them contains three integers x_i, y_i and k_i ($1 \le x_i, y_i \le n$; $0 \le k_i \le 2^{20} - 1$).

Output

For each query, print `YES` if it is possible to form a set of vertices meeting the constraints. Otherwise, print `NO`.

You can print each letter in any case.

input
4 0 1 2 10 2 1 3 2 4 2 8 3 3 0 3 4 1 3 4 7 1 3 1 1 3 2 1 3 10 1 4 10 1 4 11

output

```
YES
YES
NO
YES
YES
NO
YES
YES
```