Educational Codeforces Round 54 (Rated for Div. 2)

A. Minimizing the String

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a string s consisting of n lowercase Latin letters.

You have to remove **at most one** (i.e. zero or one) character of this string in such a way that the string you obtain will be lexicographically smallest among all strings that can be obtained using this operation.

String $s = s_1 s_2 \dots s_n$ is *lexicographically smaller* than string $t = t_1 t_2 \dots t_m$ if $n < m$ and $s_1 = t_1, s_2 = t_2, \dots, s_n = t_n$ or there exists a number p such that $p \le min(n, m)$ and $s_1 = t_1, s_2 = t_2, \dots, s_{p-1} = t_{p-1}$ and $s_p < t_p$.

For example, "aaa" is smaller than "aaaa", "abb" is smaller than "abc", "pqr" is smaller than "z".

Input

The first line of the input contains one integer n ($2 \le n \le 2 \cdot 10^5$) — the length of s.

The second line of the input contains exactly n lowercase Latin letters — the string s.

Output

Print one string — the smallest possible lexicographically string that can be obtained by removing **at most one** character from the string s.

Examples

input
```
3
aaa
```
output
```
aa
```

input
```
5
abcda
```
output
```
abca
```

Note

In the first example you can remove any character of s to obtain the string "aa".

In the second example "abca" < "abcd" < "abcda" < "abda" < "acda" < "bcda".

B. Divisor Subtraction

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an integer number n. The following algorithm is applied to it:

1. if $n = 0$, then end algorithm;
2. find the smallest **prime** divisor d of n;
3. subtract d from n and go to step 1.

Determine the number of subtrations the algorithm will make.

Input

The only line contains a single integer n ($2 \le n \le 10^{10}$).

Output

Print a single integer — the number of subtractions the algorithm will make.

Examples

input
5
output
1

input
4
output
2

Note

In the first example 5 is the smallest prime divisor, thus it gets subtracted right away to make a 0.

In the second example 2 is the smallest prime divisor at both steps.

C. Meme Problem

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Try guessing the statement from this picture:

You are given a non-negative integer d. You have to find two non-negative real numbers a and b such that $a + b = d$ and $a \cdot b = d$.

Input

The first line contains t $(1 \le t \le 10^3)$ — the number of test cases.

Each test case contains one integer d $(0 \le d \le 10^3)$.

Output

For each test print one line.

If there is an answer for the i-th test, print "Y", and then the numbers a and b.

If there is no answer for the i-th test, print "N".

Your answer will be considered correct if $|(a + b) - a \cdot b| \le 10^{-6}$ and $|(a + b) - d| \le 10^{-6}$.

Example

```
input
7
69
0
1
4
5
999
1000
```

```
output
Y 67.985071301 1.014928699
Y 0.000000000 0.000000000
N
Y 2.000000000 2.000000000
Y 3.618033989 1.381966011
Y 997.998996990 1.001003010
```

```
Y 998.998997995 1.001002005
```

D. Edge Deletion

time limit per test: 2.5 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You are given an undirected connected weighted graph consisting of n vertices and m edges. Let's denote the length of the shortest path from vertex 1 to vertex i as d_i.

You have to erase some edges of the graph so that at most k edges remain. Let's call a vertex i **good** if there still exists a path from 1 to i with length d_i after erasing the edges.

Your goal is to erase the edges in such a way that the number of **good** vertices is maximized.

Input

The first line contains three integers n, m and k ($2 \le n \le 3 \cdot 10^5$, $1 \le m \le 3 \cdot 10^5$, $n - 1 \le m$, $0 \le k \le m$) — the number of vertices and edges in the graph, and the maximum number of edges that can be retained in the graph, respectively.

Then m lines follow, each containing three integers x, y, w ($1 \le x, y \le n$, $x \ne y$, $1 \le w \le 10^9$), denoting an edge connecting vertices x and y and having weight w.

The given graph is connected (any vertex can be reached from any other vertex) and simple (there are no self-loops, and for each unordered pair of vertices there exists at most one edge connecting these vertices).

Output

In the first line print e — the number of edges that should remain in the graph ($0 \le e \le k$).

In the second line print e **distinct** integers from 1 to m — the indices of edges that should remain in the graph. Edges are numbered in the same order they are given in the input. The number of **good** vertices should be as large as possible.

Examples

input

```
3 3 2
1 2 1
3 2 1
1 3 3
```

output

```
2
1 2
```

input

```
4 5 2
4 1 8
2 4 1
2 1 3
3 4 9
3 1 5
```

output

```
2
3 2
```

E. Vasya and a Tree

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Vasya has a tree consisting of n vertices with root in vertex 1. At first all vertices has 0 written on it.

Let $d(i, j)$ be the distance between vertices i and j, i.e. number of edges in the shortest path from i to j. Also, let's denote k-subtree of vertex x — set of vertices y such that next two conditions are met:

- x is the ancestor of y (each vertex is the ancestor of itself);
- $d(x, y) \le k$.

Vasya needs you to process m queries. The i-th query is a triple v_i, d_i and x_i. For each query Vasya adds value x_i to each vertex from d_i-subtree of v_i.

Report to Vasya all values, written on vertices of the tree after processing all queries.

Input

The first line contains single integer n ($1 \le n \le 3 \cdot 10^5$) — number of vertices in the tree.

Each of next $n-1$ lines contains two integers x and y ($1 \le x, y \le n$) — edge between vertices x and y. It is guarantied that given graph is a tree.

Next line contains single integer m ($1 \le m \le 3 \cdot 10^5$) — number of queries.

Each of next m lines contains three integers v_i, d_i, x_i ($1 \le v_i \le n$, $0 \le d_i \le 10^9$, $1 \le x_i \le 10^9$) — description of the i-th query.

Output

Print n integers. The i-th integers is the value, written in the i-th vertex after processing all queries.

Examples

input

```
5
1 2
1 3
2 4
2 5
3
1 1 1
2 0 10
4 10 100
```

output

```
1 11 1 100 0
```

input

```
5
2 3
2 1
5 4
3 4
5
2 0 4
3 10 1
1 2 3
2 3 10
1 1 7
```

output

```
10 24 14 11 11
```

Note

In the first exapmle initial values in vertices are $0, 0, 0, 0, 0$. After the first query values will be equal to $1, 1, 1, 0, 0$. After the second query values will be equal to $1, 11, 1, 0, 0$. After the third query values will be equal to $1, 11, 1, 100, 0$.

F. Summer Practice Report

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vova has taken his summer practice this year and now he should write a report on how it went.

Vova has already drawn all the tables and wrote down all the formulas. Moreover, he has already decided that the report will consist of exactly n pages and the i-th page will include x_i tables and y_i formulas. The pages are numbered from 1 to n.

Vova fills the pages one after another, he can't go filling page $i+1$ before finishing page i and he can't skip pages.

However, if he draws **strictly more** than k tables in a row or writes **strictly more** than k formulas in a row then he will get bored. Vova wants to rearrange tables and formulas in each page in such a way that he doesn't get bored in the process. Vova can't move some table or some formula to another page.

Note that the count doesn't reset on the start of the new page. For example, if the page ends with 3 tables and the next page starts with 5 tables, then it's counted as 8 tables in a row.

Help Vova to determine if he can rearrange tables and formulas on each page in such a way that there is no more than k tables in a row and no more than k formulas in a row.

Input

The first line contains two integers n and k ($1 \le n \le 3 \cdot 10^5$, $1 \le k \le 10^6$).

The second line contains n integers $x_1, x_2, \dots, x_n$ ($1 \le x_i \le 10^6$) — the number of tables on the i-th page.

The third line contains n integers $y_1, y_2, \dots, y_n$ ($1 \le y_i \le 10^6$) — the number of formulas on the i-th page.

Output

Print "YES" if Vova can rearrange tables and formulas on each page in such a way that there is no more than k tables in a row and no more than k formulas in a row.

Otherwise print "NO".

Examples

input

```
2 2
5 5
2 2
```

output

```
YES
```

input

```
2 2
5 6
2 2
```

output

```
NO
```

input

```
4 1
4 1 10 1
3 2 10 1
```

output

```
YES
```

Note

In the first example the only option to rearrange everything is the following (let table be 'T' and formula be 'F'):

- page 1: "TTFTTFT"
- page 2: "TFTTFTT"

That way all blocks of tables have length 2.

In the second example there is no way to fit everything in such a way that there are no more than 2 tables in a row and 2 formulas in a row.

G. Array Game

time limit per test: 6 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Consider a following game between two players:

There is an array b_1, b_2, ..., b_k, consisting of positive integers. Initially a chip is placed into the first cell of the array, and b_1 is decreased by 1. Players move in turns. Each turn the current player has to do the following: if the index of the cell where the chip is currently placed is x, then he or she has to choose an index $y \in [x, min(k, x+m)]$ such that $b_y > 0$, move the chip to the cell y and decrease b_y by 1. If it's impossible to make a valid move, the current player loses the game.

Your task is the following: you are given an array a consisting of n positive integers, and q queries to it. There are two types of queries:

- $1\ l\ r\ d$ — for every $i \in [l, r]$ increase a_i by d;
- $2\ l\ r$ — tell who is the winner of the game that is played on the subarray of a from index l to index r inclusive. Assume both players choose an optimal strategy.

Input

The first line contains three integers n, m and q ($1 \le n, q \le 2 \cdot 10^5$, $1 \le m \le 5$) — the number of elements in a, the parameter described in the game and the number of queries, respectively.

The second line contains n integers a_1, a_2, ..., a_n ($1 \le a_i \le 10^{12}$) — the elements of array a.

Then q lines follow, each containing a query. There are two types of queries. The query of the first type is denoted by a line $1\ l\ r\ d$ ($1 \le l \le r \le n$, $1 \le d \le 10^{12}$) and means that for every $i \in [l, r]$ you should increase a_i by d. The query of the second type is denoted by a line $2\ l\ r$ ($1 \le l \le r \le n$) and means that you have to determine who will win the game if it is played on the subarray of a from index l to index r (inclusive).

There is at least one query of type 2.

Output

For each query of type 2 print 1 if the first player wins in the corresponding game, or 2 if the second player wins.

Examples

input

```
5 2 4
1 2 3 4 5
1 3 5 6
2 2 5
1 1 2 3
2 1 5
```

output

```
1
1
```

input

```
5 1 3
1 1 3 3 4
2 1 5
2 2 5
2 3 5
```

output

```
1
2
1
```