Codeforces Round #497 (Div. 2)

A. Romaji

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

Vitya has just started learning Berlanese language. It is known that Berlanese uses the Latin alphabet. Vowel letters are "a", "o", "u", "i", and "e". Other letters are consonant.

In Berlanese, there has to be a vowel after every consonant, but there can be any letter after any vowel. The only exception is a consonant "n"; after this letter, there can be any letter (not only a vowel) or there can be no letter at all. For example, the words "harakiri", "yupie", "man", and "nbo" are Berlanese while the words "horse", "king", "my", and "nz" are not.

Help Vitya find out if a word s is Berlanese.

Input
The first line of the input contains the string s consisting of $|s|$ ($1 \leq |s| \leq 100$) lowercase Latin letters.

Output
Print "YES" (without quotes) if there is a vowel after every consonant except "n", otherwise print "NO".

You can print each letter in any case (upper or lower).

Examples

input
sumimasen
output
YES

input
ninja
output
YES

input
codeforces
output
NO

Note
In the first and second samples, a vowel goes after each consonant except "n", so the word is Berlanese.

In the third sample, the consonant "c" goes after the consonant "r", and the consonant "s" stands on the end, so the word is not Berlanese.

B. Turn the Rectangles

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

There are n rectangles in a row. You can either turn each rectangle by 90 degrees or leave it as it is. If you turn a rectangle, its width will be height, and its height will be width. Notice that you can turn any number of rectangles, you also can turn all or none of them. **You can not change the order of the rectangles.**

Find out if there is a way to make the rectangles go in order of non-ascending height. In other words, after all the turns, a height of every rectangle has to be not greater than the height of the previous rectangle (if it is such).

Input
The first line contains a single integer n ($1 \leq n \leq 10^5$) — the number of rectangles.

Each of the next n lines contains two integers w_i and h_i ($1 \leq w_i, h_i \leq 10^9$) — the width and the height of the i-th rectangle.

Output
Print "YES" (without quotes) if there is a way to make the rectangles go in order of non-ascending height, otherwise print "NO".

You can print each letter in any case (upper or lower).

Examples

input
3 3 4 4 6 3 5
output
YES

input
2 3 4 5 5
output
NO

Note
In the first test, you can rotate the second and the third rectangles so that the heights will be `[4, 4, 3]`.

In the second test, there is no way the second rectangle will be not higher than the first one.

C. Reorder the Array

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

You are given an array of integers. Vasya can permute (change order) its integers. He wants to do it so that as many as possible integers will become on a place where a smaller integer used to stand. Help Vasya find the maximal number of such integers.

For instance, if we are given an array $[10, 20, 30, 40]$, we can permute it so that it becomes $[20, 40, 10, 30]$. Then on the first and the second positions the integers became larger ($20 > 10$, $40 > 20$) and did not on the third and the fourth, so for this permutation, the number that Vasya wants to maximize equals 2. Read the note for the first example, there is one more demonstrative test case.

Help Vasya to permute integers in such way that the number of positions in a new array, where integers are greater than in the original one, is maximal.

Input
The first line contains a single integer n ($1 \leq n \leq 10^5$) — the length of the array.

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

Output
Print a single integer — the maximal number of the array's elements which after a permutation will stand on the position where a smaller element stood in the initial array.

Examples

input
7 10 1 1 1 5 5 3
output
4

input
5 1 1 1 1 1
output
0

Note
In the first sample, one of the best permutations is $[1, 5, 5, 3, 10, 1, 1]$. On the positions from second to fifth the elements became larger, so the answer for this permutation is 4.

In the second sample, there is no way to increase any element with a permutation, so the answer is 0.

D. Pave the Parallelepiped

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

You are given a rectangular parallelepiped with sides of positive integer lengths A, B and C.

Find the number of different groups of three integers (a, b, c) such that $1 \le a \le b \le c$ and parallelepiped $A \times B \times C$ can be paved with parallelepipeds $a \times b \times c$. Note, that all small parallelepipeds **have to be rotated in the same direction**.

For example, parallelepiped $1 \times 5 \times 6$ can be divided into parallelepipeds $1 \times 3 \times 5$, but can not be divided into parallelepipeds $1 \times 2 \times 3$.

Input

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

Each of the next t lines contains three integers A, B and C ($1 \le A, B, C \le 10^5$) — the sizes of the parallelepiped.

Output

For each test case, print the number of different groups of three points that satisfy all given conditions.

Example

input
4 1 1 1 1 6 1 2 2 2 100 100 100
output
1 4 4 165

Note

In the first test case, rectangular parallelepiped $(1, 1, 1)$ can be only divided into rectangular parallelepiped with sizes $(1, 1, 1)$.

In the second test case, rectangular parallelepiped $(1, 6, 1)$ can be divided into rectangular parallelepipeds with sizes $(1, 1, 1)$, $(1, 1, 2)$, $(1, 1, 3)$ and $(1, 1, 6)$.

In the third test case, rectangular parallelepiped $(2, 2, 2)$ can be divided into rectangular parallelepipeds with sizes $(1, 1, 1)$, $(1, 1, 2)$, $(1, 2, 2)$ and $(2, 2, 2)$.

E. Guess two numbers

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

This is an interactive problem.

Vasya and Vitya play a game. Vasya thought of two integers a and b from 1 to n and Vitya tries to guess them. Each round he tells Vasya two numbers x and y from 1 to n. If both $x = a$ and $y = b$ then Vitya wins. Else Vasya must say one of the three phrases:

1. x is less than a;
2. y is less than b;
3. x is greater than a or y is greater than b.

Vasya can't lie, but if multiple phrases are true, he may choose any of them. For example, if Vasya thought of numbers 2 and 4, then he answers with the phrase 3 to a query $(3, 4)$, and he can answer with the phrase 1 or phrase 3 to a query $(1, 5)$.

Help Vitya win in no more than 600 rounds.

Input

The first line contains a single integer n ($1 \le n \le 10^{18}$) — the upper limit of the numbers.

Interaction

First, you need to read the number n, after that you can make queries.

To make a query, print two integers: x and y ($1 \le x, y \le n$), then `flush` the output.

After each query, read a single integer ans ($0 \le ans \le 3$).

If $ans > 0$, then it is the number of the phrase said by Vasya.

If $ans = 0$, it means that you win and your program should terminate.

If you make more than 600 queries or make an incorrect query, you will get `Wrong Answer`.

Your solution will get `Idleness Limit Exceeded`, if you don't print anything or forget to `flush` the output.

To `flush` you need to do the following right after printing a query and a line end:

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

Hacks format

For hacks, use the following format:

In the first line, print a single integer n ($1 \le n \le 10^{18}$) — the upper limit of the numbers.

In the second line, print two integers a and b ($1 \le a, b \le n$) — the numbers which Vasya thought of.

In the third line, print a single integer m ($1 \le m \le 10^5$) — the number of instructions for the interactor.

In each of the next m lines, print five integers: x_i, y_i, r_i^{12}, r_i^{13}, and r_i^{23} ($1 \le x_i, y_i \le n$), where r_i^{ST} equals to either number S or number T.

While answering the query x y, the interactor finds a number i from 1 to n with the minimal value $|x - x_i| + |y - y_i|$. If multiple numbers can be chosen, the least i is preferred. Then the interactor answers to the query, but if there are two phrases S and T that can be given, then r_i^{ST} is chosen.

For example, the sample test data file contains the following:

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

Example

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

Note

Let's analyze the sample test. The chosen numbers are 2 and 4. The interactor was given two instructions.

For the query $(4, 3)$, it can return 2 or 3. Out of the two instructions the second one is chosen, so the interactor returns $a_2^{23} = 3$.

For the query $(3, 4)$, it can return only 3.

For the query $(3, 3)$, it can return 2 or 3. Out of the two instructions the first one is chosen (since in case of equal values, the least number is preferred), so the interactor returns $a_1^{23} = 2$.

For the query $(1, 5)$, it can return 1 or 3. Out of the two instructions the first one is chosen, so the interactor returns $a_1^{13} = 1$.

In the fifth query $(2, 4)$, the numbers are guessed correctly, the player wins.