Information

Memory limit

The limit is 512 MiB for each problem.

Source code limit

The size of each solution source code can't exceed 256 KiB.

Submissions limit

You can submit at most 50 solutions for each problem.

You can submit a solution to each task at most once per 30 seconds. This restriction does not apply in the last 15 minutes of the contest round.

Scoring

Each problem consists of several subtasks. The subtask score is awarded if all tests in the subtask are passed.

The number of points scored for the problem is the total number of points scored on each of its subtasks. The score for the subtask is the maximum number of points earned for this subtask among all the solutions submitted.

Feedback

To get feedback for your solution, go to "Runs" tab in PCMS2 Web Client and use "View Feedback" link. In each problem of the contest you will see the score for each subtask, or the verdict for the first failed test.

Scoreboard

The contestants' scoreboard is available during the contest. Use "Monitor" link in PCMS2 Web Client to access the scoreboard. The standings provided in PCMS2 Web Client are not final.

Problem A. Homework

Time limit: 2 seconds

Lana and Alan are twins, and they study in the same school. One day, they were given homework in which they had to make up a word using given letters. As they didn't want to be considered cheaters, they wanted to make up words that **differ at every position**.

Given a string of lowercase English letters A, find two words B and C that use all these letters (in other words, they both must be an anagram of A), such that B and C contain different letters in each place. These words don't have to be actual words of any real-world language.

Input

The input contains a string A, consisting of lowercase English letters. The length of A is between 1 and 1000.

Output

If an answer exists, print two lines containing words B and C. Each word must be an anagram of A (containing the same letters as A, possibly, in a different order). For each i from 1 to the length of A, B_i must be different from C_i.

If it's impossible to make up such B and C, print "`IMPOSSIBLE`". If there is more than one correct answer, you can print any of them.

Scoring

Subtask	Points	Constraints
1	30	All letters in A are distinct
2	30	The length of A does not exceed 10
3	40	No additional constraints

Examples

standard input	standard output
`nala`	`alan` `lana`
`abacaba`	`IMPOSSIBLE`
`innopolisopen`	`noepsilonnopi` `opinionnpoles`

Problem B. Matryoshka Inc

Time limit: 2 seconds

A prominent toy company `Matryoshka Inc` is going to launch a marketing campaign. They will invite participants to solve the following problem: there are n matryoshkas in a row, numbered from 1 to n, the i-th matryoshka's size is s_i. The participant's goal is to assemble a stack of matryoshkas using as many matryoshkas as possible. But, there is a restriction: they must create the stack using matryoshkas in order from left to right, from the smallest to largest (possibly, skipping some of them).

To begin, you can choose the first matryoshka, then select some other matryoshka with a larger index and strictly larger size, and then put the first matryoshka inside the second one. After this, you can select another matryoshka whose size and index are strictly larger than the size and index of previously selected matryoshkas, put the first two matryoshkas inside the third one, and so on. When you are done, your score is the number of matryoshkas used. In the case when you cannot put a matryoshka inside another one with a larger index, or if there is just one matryoshka, the score is considered to be equal to one.

Now the CEO of `Matryoshka Inc` is interested in the maximum score a participant can achieve. They sent you a message with n numbers $s_1, \ldots, s_n$, but there were some connection issues. Instead of n numbers $s_1, \ldots, s_n$, you received n numbers $a_1, \ldots, a_n$, where a_i is equal to s_i, but the digits were shuffled, possibly with leading zeroes.

There is very little time, so the CEO asks you to determine the maximum score a participant can achieve based only on $a_1, \ldots, a_n$. Recall that a_i is a digit permutation of s_i, possibly with leading zeroes.

Input

The first line contains the single integer n ($1 \le n \le 1000$) — the number of matryoshkas.

The second line contains n integers $a_1, \ldots, a_n$ ($0 \le a_i < 10^{18}$). Each integer a_i has at most 18 digits, possibly, with leading zeroes.

Output

Output a single integer — the maximum possible score that a participant can theoretically achieve.

Scoring

Let's define L as the maximum length of an input number a_i.

Subtask	Points	Constraints
1	12	$n \le 3$
2	8	$n \le 18$
3	9	$L = 1$
4	17	$L \le 3$
5	26	$n \le 200, 1 \le L \le 8$
6	28	No additional constraints

Examples

standard input	standard output
2 1 1	1
5 10 2 30 4 50	5
6 77 88 91 22 33 44	4
3 390100 200200 012	3

Note

Let's consider the first example. In that case, you have only two matryoshkas of equal sizes. It means that you can not put any matryoshka inside the other one. So, the answer for the first example is 1.

Lets's consider the second example. Theoretically, the sequence of the matryoshka's sizes could be 01, 2, 03, 4, 05. In that case, a participant could achieve a score equal to 5. So, the answer for the second example is 5.

Let's consider the third example. There are only two ways the real sequence of matryoshka's sizes could look like. The first one is 77, 88, 91, 22, 33. And the second one is 77, 88, 19, 22, 33. It's easy to see that in the first case, the maximum score that the participant can achieve equals 3, but in the second case, the maximum score that the participant can achieve equals 4. So, the answer for the third example is 4.

Let's consider the fourth example. There is a scenario where the sequence of the matryoshka's sizes is 000139, 000202, 210. So, participant could assemble a stack of three matryoshkas. So, the answer for the forth example is 3.

Problem C. Password Lock

Time limit: 2 seconds

Andrey got a cool new lock for his birthday. You can set several passwords on it at the same time; when you enter any of them, the lock opens.

The lock consists of a sequence of cells, each has an integer written on it. You can swap places of any two cells any number of times; when they are arranged in any correct order, the lock opens. The order is considered correct when **the sum of two integers on any pair of adjacent cells is not divisible by k**.

You found out about the structure of this lock and, of course, decided to try to open it. Given the numbers on the cells and the value of k, find the permutation of cells that opens the lock.

Input

The first line of the input contains two integers n and k ($1 \le n \le 10^5$, $1 \le k \le 10^9$) — the number of cells on the lock and the number k.

The second line of the input contains n integers a_i ($1 \le a_i \le 10^9$) — the initial configuration of cells on the lock.

Output

If there is no suitable permutation of cells, print "`NO`" (without quotes, case insensitive) on the first line.

If a suitable permutation of cells exists, print "`YES`" (no quotes, case insensitive) on the first line. In the second line, print n integers — the permutation itself.

Scoring

Subtask	Points	Constraints
1	8	$n \le 8$
2	8	$k = 2$
3	8	$k = 3$
4	8	$n = k$; numbers a_i are distinct and are between 1 and n
5	25	k is odd
6	43	No additional restrictions

Examples

standard input	standard output
3 4 1 3 2	YES 1 2 3
5 4 1 2 2 2 4	YES 2 4 2 1 2

Problem D. The Name of the Fourth Problem

Time limit: 2 seconds

Consider an infinite sequence of integers $a_1, a_2, a_3, \ldots$. The sequence is called a *self-describing* if a_i is equal to the number of occurrences of i in this sequence.

If we restrict this sequence to satisfy $a_1 = 1$ and the numbers a non-decreasing (that is, $a_i \le a_{i+1}$ for all i), then the self-describing sequence is unique. The first thirty elements are as follows:

$$1, 2, 2, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6, 6, 7, 7, 7, 7, 8, 8, 8, 8, 9, 9, 9, 9, 9, 10, 10, \ldots$$

Your task is to calculate the range sum queries for this sequence efficiently. Let's go!

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of range sum queries. The following t lines contains two integers l_i and r_i ($1 \le l_i \le r_i \le 10^{15}$) each. The i-th query requires you to calculate $\sum_{k=l_i}^{r_i} a_k$.

Output

Print t lines, each one with a single integer — the answer to the range sum query. Because the sum can be large, print it modulo 1 000 000 007.

Scoring

Subtask	Points	Constraints
1	12	$l_i = r_i$; $r_i \le 1000$
2	7	$l_i = r_i$; $r_i \le 10^6$
3	8	$r_i \le 10^6$
4	30	$r_i \le 10^{10}$
5	43	No additional constraints

Example

standard input	standard output

5
1 1
2 2
3 4
56 56
5 13
	1
	2
	5
	14
	42

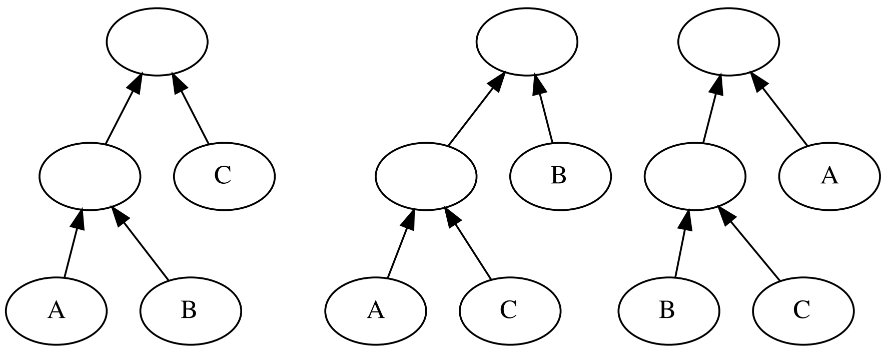
Problem E. Comparing Theories

Time limit: 3 seconds

Testing hypotheses and theories is an integral part of the scientific process. One of the most essential questions in evolutionary biology is studying the history of how the species developed; when and how the species evolution diverged into new species. In this problem, you will evaluate the difference between two evolutionary hypotheses, represented as trees.

The evolutionary history of n modern species can be represented as a binary tree with n leaves corresponding to those n species. Each internal vertex corresponds to an extinct species that, at some point in time, had diverged into two other species. The root of this tree is a hypothetical "primordial" species, a predecessor to all modern species. The leaves are numbered from 1 to n; the other vertices have numbers from $n+1$ to $2n-1$. The vertex $2n-1$ is the root of the tree.

To compare two evolutionary theories, scientists use different metrics, one of them is called "triplet distance". Consider three different modern species (leaves in the tree). Go up the evolutionary tree from those species and find two species where these three modern species diverged. For three modern species A, B, and C, there are, essentially, three different evolutionary structures (shown in the picture below): the common ancestor split into one species that then evolved into A and B, and another one, that eventually evolved into C. Similarly, there are two more structures with different partitions. We'll say that, for a given triple (A, B, C), its history is different in two theories if these basic structures differ. Shown below are the three different historic structures for the three modern species. Every edge in the picture might be a chain of multiple intermediate species.



Find the number of triples of modern species with different historic structures in the two given evolutionary trees.

Input

The first line contains one integer n ($3 \le n \le 50\,000$) — the number of modern species (also the number of leaves in the tree). Then, two lines describing the trees follow.

Each tree is represented as an array of parents of length $2n-2$: for every vertex i from 1 to $2n-2$, you are given the number of its parent in the tree `parent[`i`]` $> i$. Vertex number $2n-1$ is the root of the tree. It's guaranteed that the vertices 1 to n are leaves, and they don't appear as values in the `parent` array. It is also guaranteed that the tree is a binary tree: each internal vertex has a number from $n+1$ to $2n-1$ and has exactly two children (that is, all these values appear exactly twice in the `parent` array).

Output

Print a single integer — the number of triples of species with different evolutionary histories.

Scoring

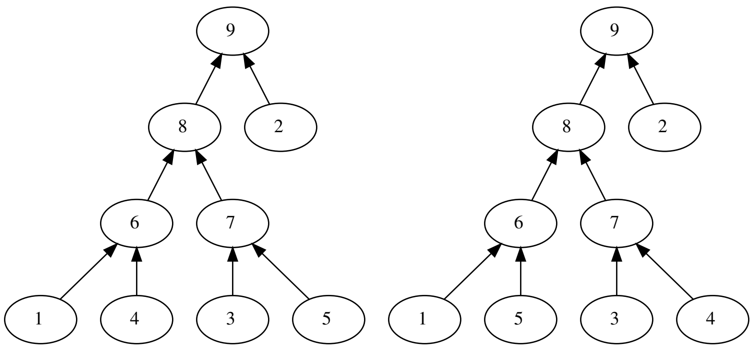
Subtask	Points	Constraints
1	25	$n \le 50$
2	10	$n \le 200$
3	20	$n \le 2000$
4	45	No additional constraints

Examples

standard input	standard output
`5`	`4`
`6 9 7 6 7 8 8 9`	
`6 9 7 7 6 8 8 9`	
`5`	`8`
`8 7 7 6 6 8 9 9`	
`7 8 6 9 6 7 8 9`	

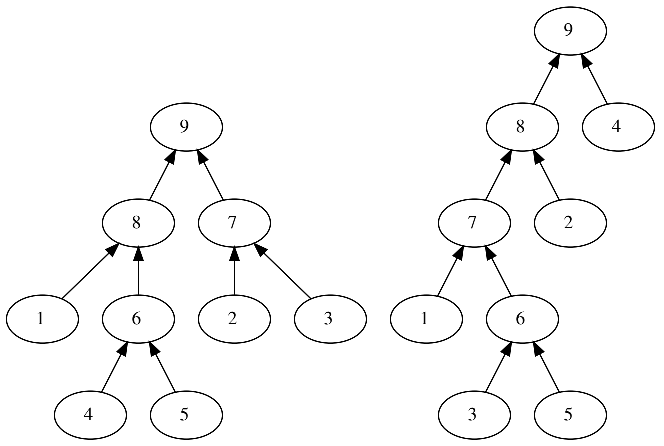
Note

These are the two trees from the first example:



Triples with different evolutionary histories: $(1, 3, 4), (1, 3, 5), (1, 4, 5), (3, 4, 5)$.

The second example:



There are 8 different triples: $(1, 2, 3), (1, 2, 4), (1, 3, 4), (1, 3, 5), (1, 4, 5), (2, 3, 5), (2, 4, 5), (3, 4, 5)$.