Educational Codeforces Round 64 (Rated for Div. 2)

A. Inscribed Figures

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

The math faculty of Berland State University has suffered the sudden drop in the math skills of enrolling students. This year the highest grade on the entrance math test was 8. Out of 100! Thus, the decision was made to make the test easier.

Future students will be asked just a single question. They are given a sequence of integer numbers $a_1, a_2, \ldots, a_n$, each number is from 1 to 3 and $a_i \neq a_{i+1}$ for each valid i. The i-th number represents a type of the i-th figure:

1. circle;
2. isosceles triangle with the length of height equal to the length of base;
3. square.

The figures of the given sequence are placed somewhere on a Cartesian plane in such a way that:

- $(i+1)$-th figure is inscribed into the i-th one;
- each triangle base is parallel to OX;
- the triangle is oriented in such a way that the vertex opposite to its base is at the top;
- each square sides are parallel to the axes;
- for each i from 2 to n figure i has the maximum possible length of side for triangle and square and maximum radius for circle.

Note that the construction is unique for some fixed position and size of just the first figure.

The task is to calculate the number of **distinct** points (not necessarily with integer coordinates) where figures touch. The trick is, however, that the number is sometimes infinite. But that won't make the task difficult for you, will it?

So can you pass the math test and enroll into Berland State University?

Input
The first line contains a single integer n ($2 \le n \le 100$) — the number of figures.

The second line contains n integer numbers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 3$, $a_i \neq a_{i+1}$) — types of the figures.

Output
The first line should contain either the word "`Infinite`" if the number of distinct points where figures touch is infinite or "`Finite`" otherwise.

If the number is finite than print it in the second line. It's guaranteed that the number fits into 32-bit integer type.

Examples

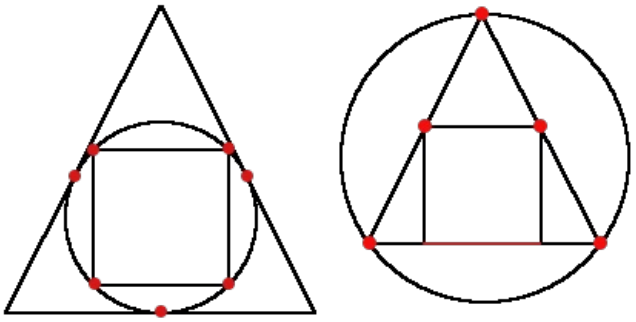
input
3 2 1 3
output
Finite 7

input
3 1 2 3
output
Infinite

Note
Here are the glorious pictures for the examples. Note that the triangle is not equilateral but just isosceles with the length of height equal to the length of base. Thus it fits into a square in a unique way.

The distinct points where figures touch are marked red.

In the second example the triangle and the square touch each other for the whole segment, it contains infinite number of points.



B. Ugly Pairs

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

You are given a string, consisting of lowercase Latin letters.

A pair of **neighbouring** letters in a string is considered ugly if these letters are also **neighbouring** in a alphabet. For example, string "abaca" contains ugly pairs at positions $(1, 2)$ — "ab" and $(2, 3)$ — "ba". Letters 'a' and 'z' aren't considered neighbouring in a alphabet.

Can you rearrange the letters of a given string so that there are no ugly pairs? You can choose any order of the letters of the given string but you can't add any new letters or remove the existing ones. You can also leave the order the same.

If there are multiple answers, print any of them.

You also have to answer T separate queries.

Input
The first line contains a single integer T ($1 \le T \le 100$) — the number of queries.

Each of the next T lines contains string s $(1 \le |s| \le 100)$ — the string for the next query. It is guaranteed that it contains only lowercase Latin letters.

Note that in hacks you have to set $T = 1$.

Output
Print T lines. The i-th line should contain the answer to the i-th query.

If the answer for the i-th query exists, then print such a rearrangment of letters of the given string that it contains no ugly pairs. You can choose any order of the letters of the given string but you can't add any new letters or remove the existing ones. You can also leave the order the same.

If there are multiple answers, print any of them.

Otherwise print "No answer" for that query.

Example

input
4 abcd gg codeforces abaca
output
cadb gg codfoerces No answer

Note
In the first example answer "bdac" is also correct.

The second example showcases the fact that only neighbouring in alphabet letters are not allowed. The same letter is ok.

There are lots of valid answers for the third example.

C. Match Points

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

output: standard output

You are given a set of points x_1, x_2, ..., x_n on the number line.

Two points i and j can be matched with each other if the following conditions hold:

- neither i nor j is matched with any other point;
- $|x_i - x_j| \geq z$.

What is the maximum number of pairs of points you can match with each other?

Input
The first line contains two integers n and z ($2 \leq n \leq 2 \cdot 10^5$, $1 \leq z \leq 10^9$) — the number of points and the constraint on the distance between matched points, respectively.

The second line contains n integers x_1, x_2, ..., x_n ($1 \leq x_i \leq 10^9$).

Output
Print one integer — the maximum number of pairs of points you can match with each other.

Examples

input
4 2 1 3 3 7
output
2

input
5 5 10 9 5 8 7
output
1

Note
In the first example, you may match point 1 with point 2 ($|3 - 1| \geq 2$), and point 3 with point 4 ($|7 - 3| \geq 2$).

In the second example, you may match point 1 with point 3 ($|5 - 10| \geq 5$).

D. 0-1-Tree

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

You are given a tree (an undirected connected acyclic graph) consisting of n vertices and $n - 1$ edges. A number is written on each edge, each number is either 0 (let's call such edges 0-edges) or 1 (those are 1-edges).

Let's call an ordered pair of vertices (x, y) ($x \neq y$) **valid** if, while traversing the simple path from x to y, we never go through a 0-edge after going through a 1-edge. Your task is to calculate the number of **valid** pairs in the tree.

Input
The first line contains one integer n ($2 \leq n \leq 200000$) — the number of vertices in the tree.

Then $n - 1$ lines follow, each denoting an edge of the tree. Each edge is represented by three integers x_i, y_i and c_i ($1 \leq x_i, y_i \leq n$, $0 \leq c_i \leq 1$, $x_i \neq y_i$) — the vertices connected by this edge and the number written on it, respectively.

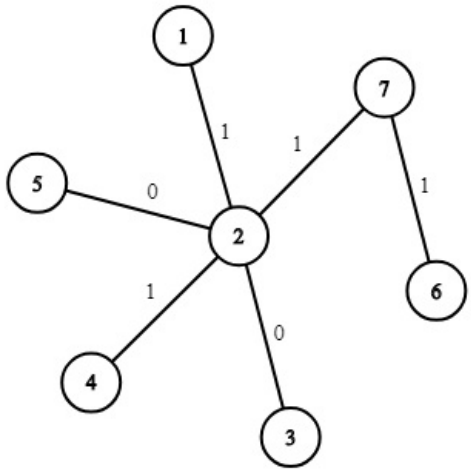
It is guaranteed that the given edges form a tree.

Output
Print one integer — the number of **valid** pairs of vertices.

Example

input
7 2 1 1 3 2 0 4 2 1 5 2 0 6 7 1 7 2 1
output
34

Note
The picture corresponding to the first example:



E. Special Segments of Permutation

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given a permutation p of n integers 1, 2, ..., n (a permutation is an array where each element from 1 to n occurs exactly once).

Let's call some subsegment $p[l, r]$ of this permutation special if $p_l + p_r = \max\limits_{i=l}^{r} p_i$. Please calculate the number of special subsegments.

Input
The first line contains one integer n ($3 \le n \le 2 \cdot 10^5$).

The second line contains n integers p_1, p_2, ..., p_n ($1 \le p_i \le n$). All these integers are pairwise distinct.

Output
Print the number of special subsegments of the given permutation.

Examples

input
5 3 4 1 5 2
output
2

input
3 1 3 2
output
1

Note
Special subsegments in the first example are $[1, 5]$ and $[1, 3]$.

The only special subsegment in the second example is $[1, 3]$.

F. Card Bag

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You have a bag which contains n cards. There is a number written on each card; the number on i-th card is a_i.

You are playing the following game. During each turn, you choose and remove a random card from the bag (all cards that are still left inside the bag are chosen equiprobably). Nothing else happens during the first turn — but during the next turns, after removing a card (let the number on it be x), you compare it with the card that was removed during the previous turn (let the number on it be y). Possible outcomes are:

- if $x < y$, the game ends and you lose;
- if $x = y$, the game ends and you win;
- if $x > y$, the game continues.

If there are no cards left in the bag, you lose. **Cards are not returned into the bag after you remove them.**

You have to calculate the probability of winning in this game. It can be shown that it is in the form of $\frac{P}{Q}$ where P and Q are non-negative integers and $Q \neq 0$, $P \leq Q$. Output the value of $P \cdot Q^{-1} \ (mod \ 998244353)$.

Input
The first line contains one integer n ($2 \leq n \leq 5000$) — the number of cards in the bag.

The second live contains n integers $a_1, a_2, \ldots a_n$ ($1 \leq a_i \leq n$) — the i-th integer is the number written on the i-th card.

Output
Print one integer — the probability of winning in this game modulo 998244353.

Examples

input
5 1 1 4 2 3
output
299473306

input
2 2 2
output
1

input
5 4 5 1 3 2
output
0

input
4 1 3 4 3
output
748683265

Note
In the first test case the probability of winning is $\frac{1}{10}$.

In the second test case the probability of winning is 1.

In the third test case the probability of winning is 0.

In the fourth test case the probability of winning is $\frac{1}{4}$.

G. Optimizer

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

Let's analyze a program written on some strange programming language. The variables in this language have names consisting of 1 to 4 characters, and each character is a lowercase or an uppercase Latin letter, or a digit. There is an extra constraint that the first character should not be a digit.

There are four types of operations in the program, each denoted by one of the characters: $, ^, # or &.

Each line of the program has one of the following formats:

- <lvalue>=<rvalue>, where <lvalue> and <rvalue> are valid variable names;
- <lvalue>=<arg1><op><arg2>, where <lvalue>, <arg1> and <arg2> are valid variable names, and <op> is an operation character.

The program is executed line-by-line, and the result of execution is stored in a variable having the name res. If res is never assigned in the program, then the result will be equal to the value of res before running the program.

Two programs are called equivalent if no matter which operations do characters $, ^, # and & denote (but, obviously, performing the same operation on the same arguments gives the same result) and which values do variables have before execution of program, the value of res after running the first program is equal to the value of res after running the second program (the programs are executed independently).

You are given a program consisting of n lines. Your task is to write a program consisting of minimum possible number of lines that is equivalent to the program you are given.

Input
The first line contains one integer n ($1 \leq n \leq 1000$) — the number of lines in the program.

Then n lines follow — the program itself. Each line corresponds to the format described in the statement and has no extra whitespaces.

Output
In the first line print k — the minimum number of lines in the equivalent program.

Then print k lines without any whitespaces — an equivalent program having exactly k lines, in the same format it is described in the statement.

Examples

input
4 c=aa#bb d12=c res=c^d12 tmp=aa$c
output
2 aaaa=aa#bb res=aaaa^aaaa

input
2 max=aaaa$bbbb min=bbbb^aaaa
output
0

Codeforces (c) Copyright 2010-2022 Mike Mirzayanov
The only programming contests Web 2.0 platform