Forethought Future Cup - Elimination Round

A. Love "A"

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

Alice has a string s. She really likes the letter "a". She calls a string good if strictly more than half of the characters in that string are "a"s. For example "aaabb", "axaa" are good strings, and "baca", "awwwa", "" (empty string) are not.

Alice can erase some characters from her string s. She would like to know what is the longest string remaining after erasing some characters (possibly zero) to get a good string. It is guaranteed that the string has at least one "a" in it, so the answer always exists.

Input
The first line contains a string s ($1 \leq |s| \leq 50$) consisting of lowercase English letters. It is guaranteed that there is at least one "a" in s.

Output
Print a single integer, the length of the longest good string that Alice can get after erasing some characters from s.

Examples

input
xaxxxxa
output
3

input
aaabaa
output
6

Note
In the first example, it's enough to erase any four of the "x"s. The answer is 3 since that is the maximum number of characters that can remain.

In the second example, we don't need to erase any characters.

B. Hate "A"

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

Bob has a string s consisting of lowercase English letters. He defines s' to be the string after removing all "a" characters from s (keeping all other characters in the same order). He then generates a new string t by concatenating s and s'. In other words, $t = s + s'$ (look at notes for an example).

You are given a string t. Your task is to find some s that Bob could have used to generate t. It can be shown that if an answer exists, it will be unique.

Input
The first line of input contains a string t ($1 \leq |t| \leq 10^5$) consisting of lowercase English letters.

Output
Print a string s that could have generated t. It can be shown if an answer exists, it is unique. If no string exists, print ":(" (without double quotes, there is no space between the characters).

Examples

input
aaaaa
output
aaaaa

input
aacaababc
output
:(

input
ababacacbbcc
output
ababacac

input
baba
output
:(

Note
In the first example, we have s = "aaaaa", and s' = "".

In the second example, no such s can work that will generate the given t.

In the third example, we have s = "ababacac", and s' = "bbcc", and $t = s + s'$ = "ababacacbbcc".

C. Tree Diameter

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There is a weighted tree with n nodes and $n - 1$ edges. The nodes are conveniently labeled from 1 to n. The weights are positive integers at most 100. Define the distance between two nodes to be the sum of edges on the unique path between the nodes. You would like to find the diameter of the tree. Diameter is the maximum distance between a pair of nodes.

Unfortunately, the tree isn't given to you, but you can ask some questions about it. In one question, you can specify two nonempty disjoint sets of nodes p and q, and the judge will return the maximum distance between a node in p and a node in q. In the words, maximum distance between x and y, where $x \in p$ and $y \in q$. After asking not more than 9 questions, you must report the maximum distance between any pair of nodes.

Interaction
Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 1\,000$). Description of the test cases follows.

The first line of each test case contains an integer n ($2 \le n \le 100$) — the number of nodes in the tree.

To ask a question, print "$k_1\ k_2\ a_1\ a_2\ \ldots\ a_{k_1}\ b_1\ b_2\ \ldots\ b_{k_2}$" ($k_1, k_2 \ge 1$ and $k_1 + k_2 \le n$). These two sets must be nonempty and disjoint. The judge will respond with a single integer $\max_{p,q} dist(a_p, b_q)$. If you ever get a result of -1 (because you printed an invalid query), exit immediately to avoid getting other verdicts.

After printing a query do not forget to output end of line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

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

When you are ready to answer, print "$-1\ d$", where d is the maximum shortest distance over all pairs of nodes.

You can only ask at most 9 questions per test case.

Hack Format

To hack, use the following format. Note that you can only hack with one test case.

The first line should contain a single integer t ($t = 1$).

The second line should contain a single integer n ($2 \le n \le 100$).

Each of the next $n-1$ lines should contain three integers a_i, b_i, c_i ($1 \le a_i, b_i \le n$, $1 \le c_i \le 100$). This denotes an undirected edge between nodes a_i and b_i with weight c_i. These edges must form a tree.

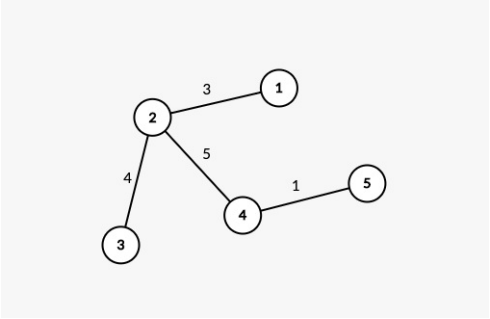
Example

input
2 5 9 6 10 9 10 2 99

output
1 4 1 2 3 4 5 1 4 2 3 4 5 1 1 4 3 4 5 1 2 1 4 4 5 1 2 3 1 4 5 1 2 3 4 -1 10 1 1 1 2 -1 99

Note

In the first example, the first tree looks as follows:



In the first question, we have $p = 1$, and $q = 2, 3, 4, 5$. The maximum distance between a node in p and a node in q is 9 (the distance between nodes 1 and 5).

The second tree is a tree with two nodes with an edge with weight 99 between them.

D. Frog Jumping

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

A frog is initially at position 0 on the number line. The frog has two positive integers a and b. From a position k, it can either jump to position $k + a$ or $k - b$.

Let $f(x)$ be the number of distinct integers the frog can reach if it never jumps on an integer outside the interval $[0, x]$. The frog doesn't need to visit all these integers in one trip, that is, an integer is counted if the frog can somehow reach it if it starts from 0.

Given an integer m, find $\sum_{i=0}^{m} f(i)$. That is, find the sum of all $f(i)$ for i from 0 to m.

Input

The first line contains three integers m, a, b ($1 \le m \le 10^9, 1 \le a, b \le 10^5$).

Output

Print a single integer, the desired sum.

Examples

input
7 5 3

output
19

input
1000000000 1 2019

output
500000001500000001

input
100 100000 1

output
101

input
6 4 5

output
10

Note

In the first example, we must find $f(0) + f(1) + \ldots + f(7)$. We have $f(0) = 1, f(1) = 1, f(2) = 1, f(3) = 1, f(4) = 1, f(5) = 3, f(6) = 3, f(7) = 8$. The sum of these values is 19.

In the second example, we have $f(i) = i + 1$, so we want to find $\sum_{i=0}^{10^9} i + 1$.

In the third example, the frog can't make any jumps in any case.

E. Hot is Cold

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

You are given an array of n integers $a_1, a_2, \ldots, a_n$.

You will perform q operations. In the i-th operation, you have a symbol s_i which is either "<" or ">" and a number x_i.

You make a new array b such that $b_j = -a_j$ if $a_j s_i x_i$ and $b_j = a_j$ otherwise (i.e. if s_i is '>', then all $a_j > x_i$ will be flipped). After doing all these replacements, a is set to be b.

You want to know what your final array looks like after all operations.

Input

The first line contains two integers n, q ($1 \leq n, q \leq 10^5$) — the number of integers and the number of queries.

The next line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^5 \leq a_i \leq 10^5$) — the numbers.

Each of the next q lines contains a character and an integer s_i, x_i. ($s_i \in \{<, >\}, -10^5 \leq x_i \leq 10^5$) - the queries.

Output

Print n integers $c_1, c_2, \ldots, c_n$ representing the array after all operations.

Examples

input
11 3 -5 -4 -3 -2 -1 0 1 2 3 4 5 > 2 > -4 < 5

output
5 4 -3 -2 -1 0 1 2 -3 4 5

input
5 5 0 1 -2 -1 2 < -2 < -1 < 0 < 1 < 2

output
0 -1 2 -1 2

Note

In the first example, the array goes through the following changes:

- Initial: $[-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5]$
- > 2: $[-5, -4, -3, -2, -1, 0, 1, 2, -3, -4, -5]$
- > -4: $[-5, -4, 3, 2, 1, 0, -1, -2, 3, -4, -5]$
- < 5: $[5, 4, -3, -2, -1, 0, 1, 2, -3, 4, 5]$

F. Leaf Partition

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

You are given a rooted tree with n nodes, labeled from 1 to n. The tree is rooted at node 1. The parent of the i-th node is p_i. A leaf is node with no children. For a given set of leaves L, let $f(L)$ denote the smallest connected subgraph that contains all leaves L.

You would like to partition the leaves such that for any two different sets x, y of the partition, $f(x)$ and $f(y)$ are disjoint.

Count the number of ways to partition the leaves, modulo 998244353. Two ways are different if there are two leaves such that they are in the same set in one way but in different sets in the other.

Input

The first line contains an integer n ($2 \leq n \leq 200\,000$) — the number of nodes in the tree.

The next line contains $n - 1$ integers $p_2, p_3, \ldots, p_n$ ($1 \leq p_i < i$).

Output

Print a single integer, the number of ways to partition the leaves, modulo 998244353.

Examples

input
5 1 1 1 1

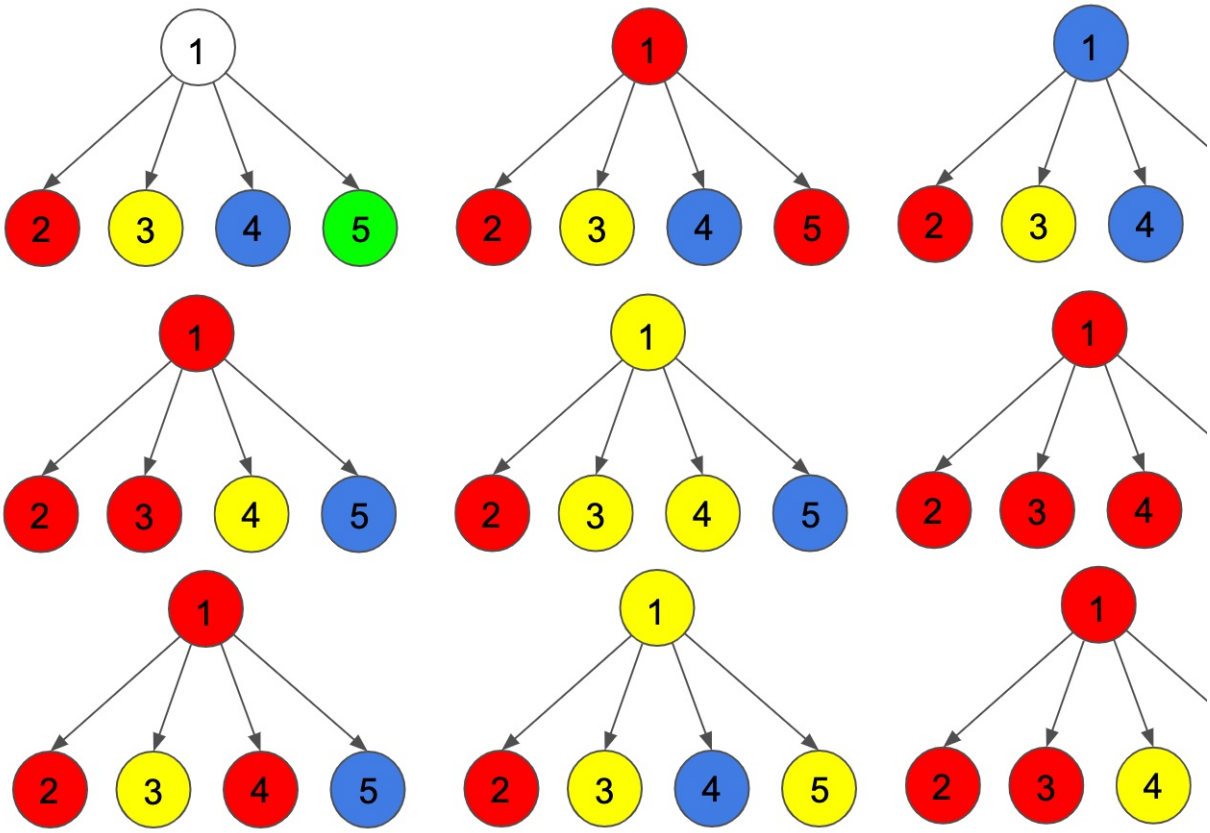
output
12

input
10 1 2 3 4 5 6 7 8 9

output
1

Note

In the first example, the leaf nodes are $2, 3, 4, 5$. The ways to partition the leaves are in the following image



In the second example, the only leaf is node 10 so there is only one partition. Note that node 1 is not a leaf.

G. Zoning Restrictions

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

You are planning to build housing on a street. There are n spots available on the street on which you can build a house. The spots are labeled from 1 to n from left to right. In each spot, you can build a house with an integer height between 0 and h.

In each spot, if a house has height a, you can gain a^2 dollars from it.

The city has m zoning restrictions though. The i-th restriction says that if the tallest house from spots l_i to r_i is strictly more than x_i, you must pay a fine of c_i.

You would like to build houses to maximize your profit (sum of dollars gained minus fines). Determine the maximum profit possible.

Input

The first line contains three integers n, h, m ($1 \leq n, h, m \leq 50$) — the number of spots, the maximum height, and the number of restrictions, respectively.

Each of the next m lines contains four integers l_i, r_i, x_i, c_i ($1 \leq l_i \leq r_i \leq n$, $0 \leq x_i \leq h$, $1 \leq c_i \leq 5\,000$).

Output

Print a single integer denoting the maximum profit you can make.

Examples

input
3 3 3 1 1 1 1000 2 2 3 1000 3 3 2 1000
output
14

input
4 10 2 2 3 8 76 3 4 7 39
output
289

Note

In the first example, it's optimal to build houses with heights $[1, 3, 2]$. We get a gain of $1^2 + 3^2 + 2^2 = 14$. We don't violate any restrictions, so there are no fees, so the total profit is $14 - 0 = 14$.

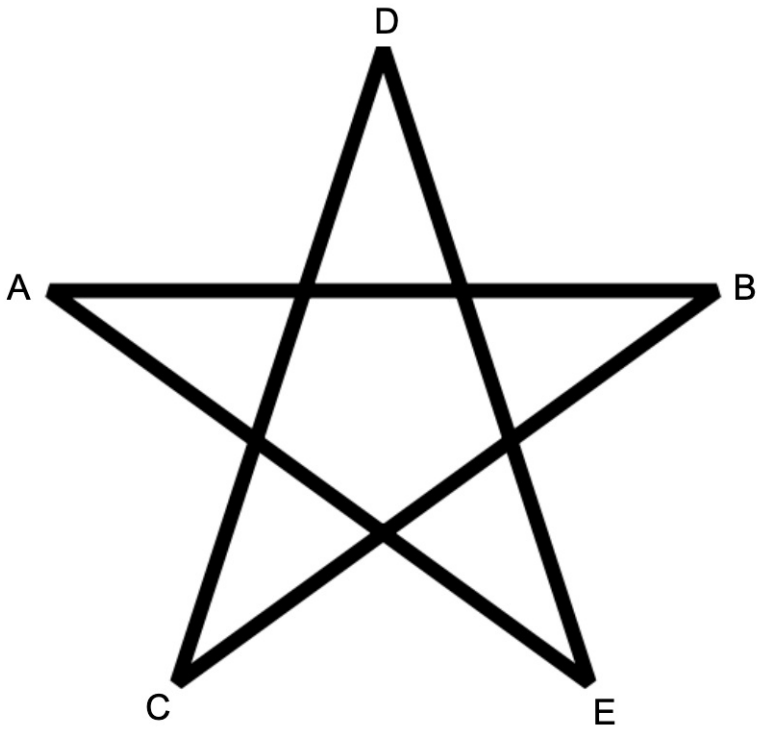
In the second example, it's optimal to build houses with heights $[10, 8, 8, 10]$. We get a gain of $10^2 + 8^2 + 8^2 + 10^2 = 328$, and we violate the second restriction for a fee of 39, thus the total profit is $328 - 39 = 289$. Note that even though there isn't a restriction on building 1, we must still limit its height to be at most 10.

H. Satanic Panic

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a set of n points in a 2D plane. No three points are collinear.

A pentagram is a set of 5 points A, B, C, D, E that can be arranged as follows.



Note the length of the line segments don't matter, only that those particular intersections exist.

Count the number of ways to choose 5 points from the given set that form a pentagram.

Input
The first line contains an integer n ($5 \le n \le 300$) — the number of points.

Each of the next n lines contains two integers x_i, y_i ($-10^6 \le x_i, y_i \le 10^6$) — the coordinates of the i-th point. It is guaranteed that no three points are collinear.

Output
Print a single integer, the number of sets of 5 points that form a pentagram.

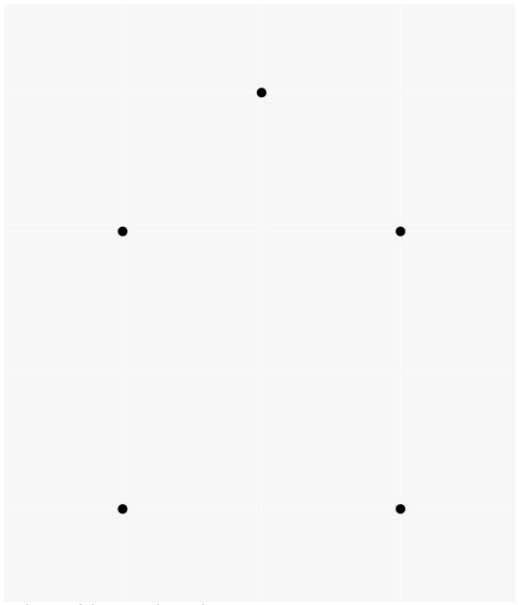
Examples

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

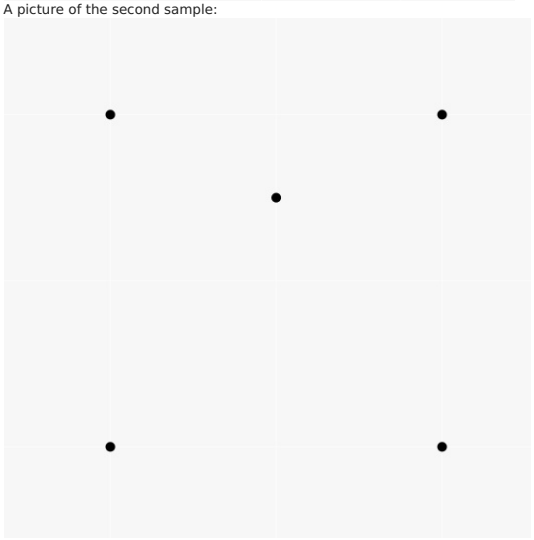
input
5 0 0 4 0 0 4 4 4 2 3
output
0

input
10 841746 527518 595261 331297 -946901 129987 670374 -140388 -684770 309555 -302589 415564 -387435 613331 -624940 -95922 945847 -199224 24636 -565799
output
85

Note
A picture of the first sample:



A picture of the second sample:



A picture of the third sample: