Codeforces Round #418 (Div. 2)

A. An abandoned sentiment from past

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

A few years ago, Hitagi encountered a giant crab, who stole the whole of her body weight. Ever since, she tried to avoid contact with others, for fear that this secret might be noticed.

To get rid of the oddity and recover her weight, a special integer sequence is needed. Hitagi's sequence has been broken for a long time, but now Kaiki provides an opportunity.

Hitagi's sequence a has a length of n. Lost elements in it are denoted by zeros. Kaiki provides another sequence b, whose length k equals the number of lost elements in a (i.e. the number of zeros). Hitagi is to replace each zero in a with an element from b so that **each element in b should be used exactly once**. Hitagi knows, however, that, **apart from 0, no integer occurs in a and b more than once in total.**

If the resulting sequence is **not** an increasing sequence, then it has the power to recover Hitagi from the oddity. You are to determine whether this is possible, or Kaiki's sequence is just another fake. In other words, you should detect whether it is possible to replace each zero in a with an integer from b so that each integer from b is used exactly once, and the resulting sequence is **not** increasing.

Input

The first line of input contains two space-separated positive integers n ($2 \le n \le 100$) and k ($1 \le k \le n$) — the lengths of sequence a and b respectively.

The second line contains n space-separated integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 200$) — Hitagi's broken sequence with exactly k zero elements.

The third line contains k space-separated integers $b_1, b_2, \ldots, b_k$ ($1 \le b_i \le 200$) — the elements to fill into Hitagi's sequence.

Input guarantees that apart from 0, no integer occurs in a and b more than once in total.

Output

Output "`Yes`" if it's possible to replace zeros in a with elements in b and make the resulting sequence not increasing, and "`No`" otherwise.

Examples

input
4 2 11 0 0 14 5 4
output
Yes

input
6 1 2 3 0 8 9 10 5
output
No

input
4 1 8 94 0 4 89
output
Yes

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

Note

In the first sample:

- Sequence a is $11, 0, 0, 14$.
- Two of the elements are lost, and the candidates in b are 5 and 4.
- There are two possible resulting sequences: $11, 5, 4, 14$ and $11, 4, 5, 14$, both of which fulfill the requirements. Thus the answer is "`Yes`".

In the second sample, the only possible resulting sequence is $2, 3, 5, 8, 9, 10$, which is an increasing sequence and therefore invalid.

B. An express train to reveries

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

Sengoku still remembers the mysterious "colourful meteoroids" she discovered with Lala-chan when they were little. In particular, one of the nights impressed her deeply, giving her the illusion that all her fancies would be realized.

On that night, Sengoku constructed a permutation $p_1, p_2, \ldots, p_n$ of integers from 1 to n inclusive, with each integer representing a colour, wishing for the colours to see in the coming meteor outburst. Two incredible outbursts then arrived, each with n meteorids, colours of which being integer sequences $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$ respectively. Meteoroids' colours were also between 1 and n inclusive, and the two sequences were not identical, that is, at least one i ($1 \le i \le n$) exists, such that $a_i \neq b_i$ holds.

Well, she almost had it all — each of the sequences a and b matched exactly $n - 1$ elements in Sengoku's permutation. In other words, there is exactly one i ($1 \le i \le n$) such that $a_i \neq p_i$, and exactly one j ($1 \le j \le n$) such that $b_j \neq p_j$.

For now, Sengoku is able to recover the actual colour sequences a and b through astronomical records, but her wishes have been long forgotten. You are to reconstruct any possible permutation Sengoku could have had on that night.

Input

The first line of input contains a positive integer n ($2 \le n \le 1\,000$) — the length of Sengoku's permutation, being the length of both meteor outbursts at the same time.

The second line contains n space-separated integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$) — the sequence of colours in the first meteor outburst.

The third line contains n space-separated integers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le n$) — the sequence of colours in the second meteor outburst. At least one i ($1 \le i \le n$) exists, such that $a_i \neq b_i$ holds.

Output

Output n space-separated integers $p_1, p_2, \ldots, p_n$, denoting a possible permutation Sengoku could have had. If there are more than one possible answer, output any one of them.

Input guarantees that such permutation exists.

Examples

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

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

input
4 1 1 3 4 1 4 3 4
output
1 2 3 4

Note

In the first sample, both $1, 2, 5, 4, 3$ and $1, 2, 3, 4, 5$ are acceptable outputs.

In the second sample, $5, 4, 2, 3, 1$ is the only permutation to satisfy the constraints.

C. An impassioned circulation of affection

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

Nadeko's birthday is approaching! As she decorated the room for the party, a long garland of Dianthus-shaped paper pieces was placed on a prominent part of the wall. Brother Koyomi will like it!

Still unsatisfied with the garland, Nadeko decided to polish it again. The garland has n pieces numbered from 1 to n from left to right, and the i-th piece has a colour s_i, denoted by a lowercase English letter. Nadeko will repaint **at most** m of the pieces to give each of them an arbitrary new colour (still denoted by a lowercase English letter). After this work, she finds out all subsegments of the garland containing pieces of only colour c — Brother Koyomi's favourite one, and takes the length of the longest among them to be the *Koyomity* of the garland.

For instance, let's say the garland is represented by "kooomo", and Brother Koyomi's favourite colour is "o". Among all subsegments containing pieces of "o" only, "ooo" is the longest, with a length of 3. Thus the *Koyomity* of this garland equals 3.

But problem arises as Nadeko is unsure about Brother Koyomi's favourite colour, and has swaying ideas on the amount of work to do. She has q plans on this, each of which can be expressed as a pair of an integer m_i and a lowercase letter c_i, meanings of which are explained above. You are to find out the maximum *Koyomity* achievable after repainting the garland according to each plan.

Input

The first line of input contains a positive integer n ($1 \le n \le 1\,500$) — the length of the garland.

The second line contains n lowercase English letters $s_1 s_2 \ldots s_n$ as a string — the initial colours of paper pieces on the garland.

The third line contains a positive integer q ($1 \le q \le 200\,000$) — the number of plans Nadeko has.

The next q lines describe one plan each: the i-th among them contains an integer m_i ($1 \le m_i \le n$) — the maximum amount of pieces to repaint, followed by a space, then by a lowercase English letter c_i — Koyomi's possible favourite colour.

Output

Output q lines: for each work plan, output one line containing an integer — the largest *Koyomity* achievable after repainting the garland according to it.

Examples

input
6 koyomi 3 1 o 4 o 4 m

output
3 6 5

input
15 yamatonadeshiko 10 1 a 2 a 3 a 4 a 5 a 1 b 2 b 3 b 4 b 5 b

output
3 4 5 7 8 1 2 3 4 5

input
10

```
aaaaaaaaaa
2
10 b
10 z
```

output

```
10
10
```

Note

In the first sample, there are three plans:

- In the first plan, at most 1 piece can be repainted. Repainting the "y" piece to become "o" results in "kooomi", whose *Koyomity* of 3 is the best achievable;
- In the second plan, at most 4 pieces can be repainted, and "oooooo" results in a *Koyomity* of 6;
- In the third plan, at most 4 pieces can be repainted, and "mmmmmi" and "kmmmmm" both result in a *Koyomity* of 5.

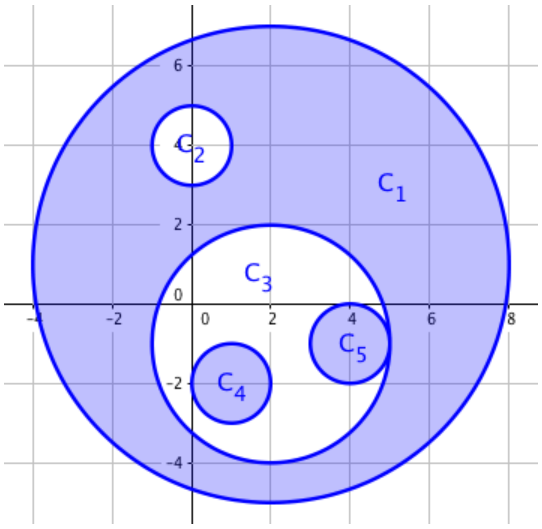
D. An overnight dance in discotheque

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

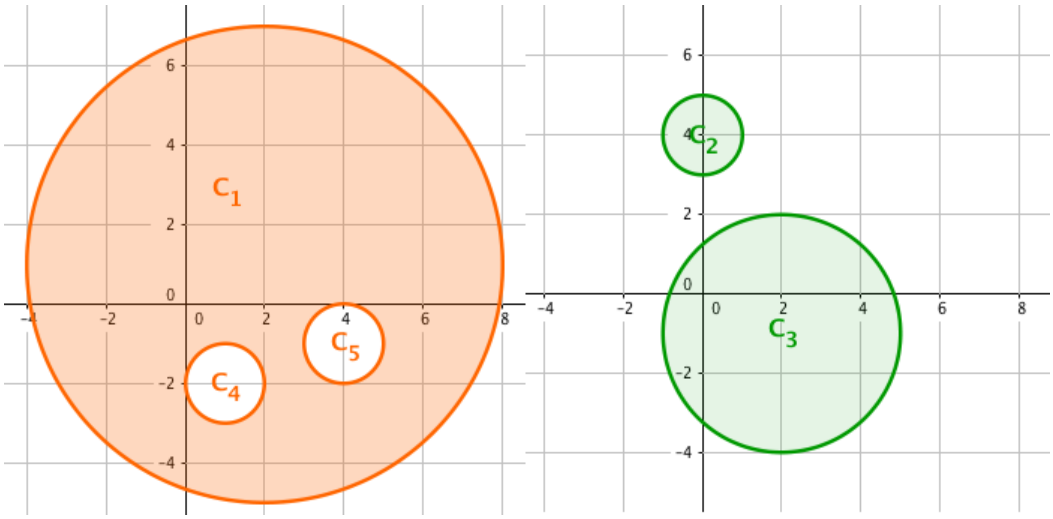
The crowdedness of the discotheque would never stop our friends from having fun, but a bit more spaciousness won't hurt, will it?

The discotheque can be seen as an infinite xy-plane, in which there are a total of n dancers. Once someone starts moving around, they will move only inside their own movement range, which is a circular area C_i described by a center (x_i, y_i) and a radius r_i. **No two ranges' borders have more than one common point**, that is for every pair (i, j) $(1 \le i < j \le n)$ either ranges C_i and C_j are disjoint, or one of them is a subset of the other. Note that it's possible that two ranges' borders share a single common point, but no two dancers have exactly the same ranges.

Tsukihi, being one of them, defines the *spaciousness* to be **the area covered by an odd number of movement ranges of dancers who are moving**. An example is shown below, with shaded regions representing the *spaciousness* if everyone moves at the same time.



But no one keeps moving for the whole night after all, so the whole night's time is divided into two halves — before midnight and after midnight. Every dancer moves around in one half, while sitting down with friends in the other. The *spaciousness* of two halves are calculated separately and their sum should, of course, be as large as possible. The following figure shows an optimal solution to the example above.



By different plans of who dances in the first half and who does in the other, different sums of *spaciousness* over two halves are achieved. You are to find the largest achievable value of this sum.

Input

The first line of input contains a positive integer n $(1 \le n \le 1\,000)$ — the number of dancers.

The following n lines each describes a dancer: the i-th line among them contains three space-separated integers x_i, y_i and r_i ($-10^6 \le x_i, y_i \le 10^6$, $1 \le r_i \le 10^6$), describing a circular movement range centered at (x_i, y_i) with radius r_i.

Output

Output one decimal number — the largest achievable sum of *spaciousness* over two halves of the night.

The output is considered correct if it has a relative or absolute error of at most 10^{-9}. Formally, let your answer be a, and the jury's answer be b. Your answer is considered correct if $\frac{|a-b|}{\max(1,|b|)} \le 10^{-9}$.

Examples

input

```
5
2 1 6
0 4 1
2 -1 3
1 -2 1
4 -1 1
```

output

```
138.23007676
```

input

```
8
0 0 1
0 0 2
0 0 3
0 0 4
0 0 5
0 0 6
0 0 7
0 0 8
```

output

```
289.02652413
```

Note

The first sample corresponds to the illustrations in the legend.

E. An unavoidable detour for home

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Those unwilling to return home from a long journey, will be affected by the oddity of the snail and lose their way. Mayoi, the oddity's carrier, wouldn't like this to happen, but there's nothing to do with this before a cure is figured out. For now, she would only like to know the enormous number of possibilities to be faced with if someone gets lost.

There are n towns in the region, numbered from 1 to n. The town numbered 1 is called the capital. The traffic network is formed by bidirectional roads connecting pairs of towns. No two roads connect the same pair of towns, and no road connects a town with itself. The time needed to travel through each of the roads is the same. Lost travelers will not be able to find out how the towns are connected, but the residents can help them by providing the following facts:

- Starting from each town other than the capital, the shortest path (i.e. the path passing through the minimum number of roads) to the capital exists, and is unique;
- Let l_i be the number of roads on the shortest path from town i to the capital, then $l_i \geq l_{i-1}$ holds for all $2 \leq i \leq n$;
- For town i, the number of roads connected to it is denoted by d_i, which equals either 2 or 3.

You are to count the number of different ways in which the towns are connected, and give the answer modulo $10^9 + 7$. Two ways of connecting towns are considered different if a pair (u, v) $(1 \leq u, v \leq n)$ exists such there is a road between towns u and v in one of them but not in the other.

Input

The first line of input contains a positive integer n $(3 \leq n \leq 50)$ — the number of towns.

The second line contains n space-separated integers $d_1, d_2, \ldots, d_n$ $(2 \leq d_i \leq 3)$ — the number of roads connected to towns $1, 2, \ldots, n$, respectively. It is guaranteed that the sum of d_i over all i is even.

Output

Output one integer — the total number of different possible ways in which the towns are connected, modulo $10^9 + 7$.

Examples

input
4 3 2 3 2
output
1

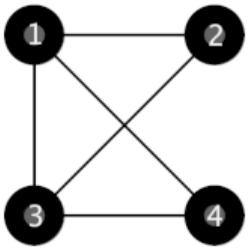
input
5 2 3 3 2 2
output
2

input
5 2 2 2 2 2
output
2

input
20 2 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 2 3 3 2
output
82944

Note

In the first example, the following structure is the only one to satisfy the constraints, the distances from towns $2, 3, 4$ to the capital are all 1.



In the second example, the following two structures satisfy the constraints.