Codeforces Round #564 (Div. 2)

A. Nauuo and Votes

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

Nauuo is a girl who loves writing comments.

One day, she posted a comment on Codeforces, wondering whether she would get upvotes or downvotes.

It's known that there were x persons who would upvote, y persons who would downvote, and there were also another z persons who would vote, but you don't know whether they would upvote or downvote. Note that each of the $x+y+z$ people would vote exactly one time.

There are three different results: if there are more people upvote than downvote, the result will be "+"; if there are more people downvote than upvote, the result will be "-"; otherwise the result will be "0".

Because of the z unknown persons, the result may be uncertain (i.e. there are more than one possible results). More formally, the result is uncertain if and only if there exist two different situations of how the z persons vote, that the results are different in the two situations.

Tell Nauuo the result or report that the result is uncertain.

Input

The only line contains three integers x, y, z ($0 \le x, y, z \le 100$), corresponding to the number of persons who would upvote, downvote or unknown.

Output

If there is **only one** possible result, print the result : "+", "-" or "0".

Otherwise, print "?" to report that the result is uncertain.

Examples

input
```
3 7 0
```
output
```
-
```

input
```
2 0 1
```
output
```
+
```

input
```
1 1 0
```
output
```
0
```

input
```
0 0 1
```
output
```
?
```

Note

In the first example, Nauuo would definitely get three upvotes and seven downvotes, so the only possible result is "-".

In the second example, no matter the person unknown downvotes or upvotes, Nauuo would get more upvotes than downvotes. So the only possible result is "+".

In the third example, Nauuo would definitely get one upvote and one downvote, so the only possible result is "0".

In the fourth example, if the only one person upvoted, the result would be "+", otherwise, the result would be "-". There are two possible results, so the result is uncertain.

B. Nauuo and Chess

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

Nauuo is a girl who loves playing chess.

One day she invented a game by herself which needs n chess pieces to play on a $m \times m$ chessboard. The rows and columns are numbered from 1 to m. We denote a cell on the intersection of the r-th row and c-th column as (r, c).

The game's goal is to place n chess pieces numbered from 1 to n on the chessboard, the i-th piece lies on $(r_i,\, c_i)$, while the following rule is satisfied: for all pairs of pieces i and j, $|r_i - r_j| + |c_i - c_j| \ge |i - j|$. Here $|x|$ means the absolute value of x.

However, Nauuo discovered that sometimes she couldn't find a solution because the chessboard was too small.

She wants to find the **smallest** chessboard on which she can put n pieces according to the rules.

She also wonders how to place the pieces on such a chessboard. Can you help her?

Input

The only line contains a single integer n ($1 \le n \le 1000$) — the number of chess pieces for the game.

Output

The first line contains a single integer — the minimum value of m, where m is the length of sides of the suitable chessboard.

The i-th of the next n lines contains two integers r_i and c_i ($1 \le r_i, c_i \le m$) — the coordinates of the i-th chess piece.

If there are multiple answers, print any.

Examples

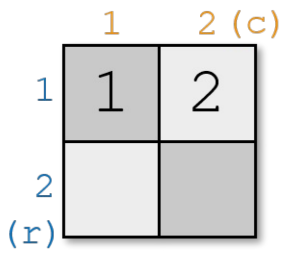
input
```
2
```
output
```
2
1 1
1 2
```

input
```
4
```
output
```
3
1 1
```

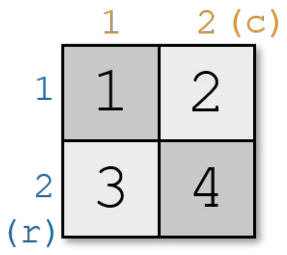
```
1 3
3 1
3 3
```

Note

In the first example, you can't place the two pieces on a 1×1 chessboard without breaking the rule. But you can place two pieces on a 2×2 chessboard like this:

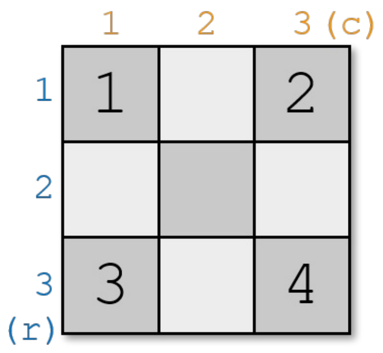


In the second example, you can't place four pieces on a 2×2 chessboard without breaking the rule. For example, if you place the pieces like this:



then $|r_1 - r_3| + |c_1 - c_3| = |1-2| + |1-1| = 1$, $|1-3| = 2$, $1 < 2$; and $|r_1 - r_4| + |c_1 - c_4| = |1-2| + |1-2| = 2$, $|1-4| = 3$, $2 < 3$. It doesn't satisfy the rule.

However, on a 3×3 chessboard, you can place four pieces like this:



C. Nauuo and Cards

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Nauuo is a girl who loves playing cards.

One day she was playing cards but found that the cards were mixed with some empty ones.

There are n cards numbered from 1 to n, and they were mixed with another n empty cards. She piled up the $2n$ cards and drew n of them. The n cards in Nauuo's hands are given. The remaining n cards in the pile are also given in the order from top to bottom.

In one operation she can choose a card in her hands and play it — put it at the bottom of the pile, then draw the top card from the pile.

Nauuo wants to make the n numbered cards piled up in increasing order (the i-th card in the pile from top to bottom is the card i) as quickly as possible. Can you tell her the minimum number of operations?

Input

The first line contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the number of numbered cards.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le n$) — the initial cards in Nauuo's hands. 0 represents an empty card.

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i \le n$) — the initial cards in the pile, given in order from top to bottom. 0 represents an empty card.

It is guaranteed that each number from 1 to n appears exactly once, either in $a_{1..n}$ or $b_{1..n}$.

Output

The output contains a single integer — the minimum number of operations to make the n numbered cards piled up in increasing order.

Examples

input
```
3
0 2 0
3 0 1
```
output
```
2
```

input
```
3
0 2 0
1 0 3
```
output
```
4
```

input

```
11
0 0 0 5 0 0 0 4 0 0 11
9 2 6 0 8 1 7 0 3 0 10
```

output

```
18
```

Note

Example 1

We can play the card 2 and draw the card 3 in the first operation. After that, we have $[0,3,0]$ in hands and the cards in the pile are $[0,1,2]$ from top to bottom.

Then, we play the card 3 in the second operation. The cards in the pile are $[1,2,3]$, in which the cards are piled up in increasing order.

Example 2

Play an empty card and draw the card 1, then play 1, 2, 3 in order.

D. Nauuo and Circle

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

Nauuo is a girl who loves drawing circles.

One day she has drawn a circle and wanted to draw a tree on it.

The tree is a connected undirected graph consisting of n nodes and $n-1$ edges. The nodes are numbered from 1 to n.

Nauuo wants to draw a tree on the circle, the nodes of the tree should be in n **distinct** points on the circle, and the edges should be straight without crossing each other.

"Without crossing each other" means that every two edges have no common point or the only common point is an endpoint of both edges.

Nauuo wants to draw the tree using a permutation of n elements. A permutation of n elements is a sequence of integers $p_1,p_2,\ldots,p_n$ in which every integer from 1 to n appears exactly once.

After a permutation is chosen Nauuo draws the i-th node in the p_i-th point on the circle, then draws the edges connecting the nodes.

The tree is given, Nauuo wants to know how many permutations are there so that the tree drawn satisfies the rule (the edges are straight without crossing each other). She only wants to know the answer modulo 998244353, can you help her?

It is obvious that whether a permutation is valid or not does not depend on which n points on the circle are chosen.

Input

The first line contains a single integer n ($2 \le n \le 2\cdot 10^5$) — the number of nodes in the tree.

Each of the next $n-1$ lines contains two integers u and v ($1 \le u,v \le n$), denoting there is an edge between u and v.

It is guaranteed that the given edges form a tree.

Output

The output contains a single integer — the number of permutations suitable to draw the given tree on a circle satisfying the rule, modulo 998244353.

Examples

input

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

output

```
16
```

input

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

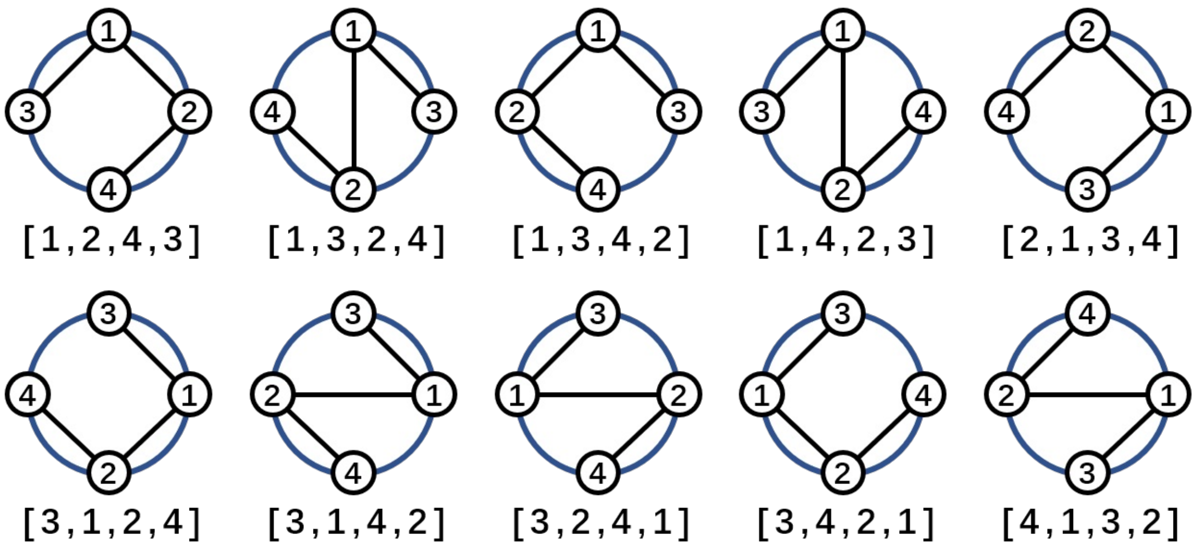
output

```
24
```

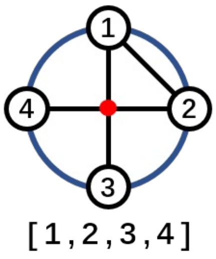
Note

Example 1

All valid permutations and their spanning trees are as follows.



Here is an example of invalid permutation: the edges $(1,3)$ and $(2,4)$ are crossed.



Example 2

Every permutation leads to a valid tree, so the answer is $4! = 24$.

E1. Nauuo and Pictures (easy version)

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

The only difference between easy and hard versions is constraints.

Nauuo is a girl who loves random picture websites.

One day she made a random picture website by herself which includes n pictures.

When Nauuo visits the website, she sees exactly one picture. The website does not display each picture with equal probability. The i-th picture has a non-negative weight w_i, and the probability of the i-th picture being displayed is $\frac{w_i}{\sum_{j=1}^n w_j}$. That is to say, the probability of a picture to be displayed is proportional to its weight.

However, Nauuo discovered that some pictures she does not like were displayed too often.

To solve this problem, she came up with a great idea: when she saw a picture she likes, she would add 1 to its weight; otherwise, she would subtract 1 from its weight.

Nauuo will visit the website m times. She wants to know the expected weight of each picture after all the m visits modulo 998244353. Can you help her?

The expected weight of the i-th picture can be denoted by $\frac{q_i}{p_i}$ where $\gcd(p_i, q_i) = 1$, you need to print an integer r_i satisfying $0 \le r_i < 998244353$ and $r_i \cdot p_i \equiv q_i \pmod{998244353}$. It can be proved that such r_i exists and is unique.

Input

The first line contains two integers n and m ($1 \le n \le 50$, $1 \le m \le 50$) — the number of pictures and the number of visits to the website.

The second line contains n integers $a_1, a_2, \ldots, a_n$ (a_i is either 0 or 1) — if $a_i = 0$, Nauuo does not like the i-th picture; otherwise Nauuo likes the i-th picture. It is guaranteed that there is at least one picture which Nauuo likes.

The third line contains n integers $w_1, w_2, \ldots, w_n$ ($1 \le w_i \le 50$) — the initial weights of the pictures.

Output

The output contains n integers $r_1, r_2, \ldots, r_n$ — the expected weights modulo 998244353.

Examples

input
```
2 1
0 1
2 1
```
output
```
332748119
332748119
```

input
```
1 2
1
1
```
output
```
3
```

input
```
3 3
0 1 1
4 3 5
```
output
```
160955686
185138929
974061117
```

Note

In the first example, if the only visit shows the first picture with a probability of $\frac{2}{3}$, the final weights are $(1, 1)$; if the only visit shows the second picture with a probability of $\frac{1}{3}$, the final weights are $(2, 2)$.

So, both expected weights are $\frac{2}{3} \cdot 1 + \frac{1}{3} \cdot 2 = \frac{4}{3}$.

Because $332748119 \cdot 3 \equiv 4 \pmod{998244353}$, you need to print 332748119 **instead of** $\frac{4}{3}$ or 1.3333333333.

In the second example, there is only one picture which Nauuo likes, so every time Nauuo visits the website, w_1 will be increased by 1.

So, the expected weight is $1 + 2 = 3$.

Nauuo is very naughty so she didn't give you any hint of the third example.

E2. Nauuo and Pictures (hard version)

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

The only difference between easy and hard versions is constraints.

Nauuo is a girl who loves random picture websites.

One day she made a random picture website by herself which includes n pictures.

When Nauuo visits the website, she sees exactly one picture. The website does not display each picture with equal probability. The i-th picture has a non-negative weight w_i, and the probability of the i-th picture being displayed is $\frac{w_i}{\sum_{j=1}^n w_j}$. That is to say, the probability of a picture to be displayed is proportional to its weight.

However, Nauuo discovered that some pictures she does not like were displayed too often.

To solve this problem, she came up with a great idea: when she saw a picture she likes, she would add 1 to its weight; otherwise, she would subtract 1 from its weight.

Nauuo will visit the website m times. She wants to know the expected weight of each picture after all the m visits modulo 998244353. Can you help her?

The expected weight of the i-th picture can be denoted by $\frac{q_i}{p_i}$ where $\gcd(p_i, q_i) = 1$, you need to print an integer r_i satisfying $0 \le r_i < 998244353$ and $r_i \cdot p_i \equiv q_i \pmod{998244353}$. It can be proved that such r_i exists and is unique.

Input

The first line contains two integers n and m ($1 \le n \le 2 \cdot 10^5$, $1 \le m \le 3000$) — the number of pictures and the number of visits to the website.

The second line contains n integers $a_1, a_2, \ldots, a_n$ (a_i is either 0 or 1) — if $a_i = 0$, Nauuo does not like the i-th picture; otherwise Nauuo likes the i-th picture. It is guaranteed that there is at least one picture which Nauuo likes.

The third line contains n positive integers $w_1, w_2, \ldots, w_n$ ($w_i \ge 1$) — the initial weights of the pictures. It is guaranteed that the sum of all the initial weights does not exceed $998244352 - m$.

Output

The output contains n integers $r_1, r_2, \ldots, r_n$ — the expected weights modulo 998244353.

Examples

input
```
2 1
0 1
2 1
```
output
```
332748119
332748119
```

input
```
1 2
1
1
```
output
```
3
```

input
```
3 3
0 1 1
4 3 5
```
output
```
160955686
185138929
974061117
```

Note

In the first example, if the only visit shows the first picture with a probability of $\frac{2}{3}$, the final weights are $(1,1)$; if the only visit shows the second picture with a probability of $\frac{1}{3}$, the final weights are $(2,2)$.

So, both expected weights are $\frac{2}{3} \cdot 1 + \frac{1}{3} \cdot 2 = \frac{4}{3}$.

Because $332748119 \cdot 3 \equiv 4 \pmod{998244353}$, you need to print 332748119 **instead of** $\frac{4}{3}$ or 1.3333333333.

In the second example, there is only one picture which Nauuo likes, so every time Nauuo visits the website, w_1 will be increased by 1.

So, the expected weight is $1 + 2 = 3$.

Nauuo is very naughty so she didn't give you any hint of the third example.

F. Nauuo and Portals

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

Nauuo is a girl who loves playing games related to portals.

One day she was playing a game as follows.

In an $n \times n$ grid, the rows are numbered from 1 to n from top to bottom, the columns are numbered from 1 to n from left to right. We denote a cell on the intersection of the r-th row and c-th column as (r, c).

A portal is **a pair of** doors. You can travel from one of them to another without changing your direction. More formally, if you *walk into* a cell with a door, you will teleport to the cell with the other door of the same portal and then *walk into* the *next cell* facing the original direction. There **can not** be more than one doors in a single cell.

The "*next cell*" is the nearest cell in the direction you are facing. For example, if you are facing bottom, the *next cell* of $(2,5)$ is $(3,5)$.

If you *walk into* a cell without a door, you must *walk into* the *next cell* after that without changing the direction. If the *next cell* does not exist, you must exit the grid.

You have to set some (possibly zero) portals in the grid, so that if you *walk into* $(i,1)$ facing right, you will eventually exit the grid from (r_i, n), if you *walk into* $(1, i)$ facing bottom, you will exit the grid from (n, c_i).

It is guaranteed that both $r_{1..n}$ and $c_{1..n}$ are **permutations** of n elements. A permutation of n elements is a sequence of numbers $p_1, p_2, \ldots, p_n$ in which every integer from 1 to n appears exactly once.

She got confused while playing the game, can you help her to find a solution?

Input

The first line contains a single integer n ($1 \le n \le 1000$) — the side length of the grid.

The second line contains n integers $r_1, r_2, \ldots, r_n$ ($1 \le r_i \le n$) — if you walk into $(i, 1)$ facing right, you should exit the grid from (r_i, n). It is guaranteed that $r_{1..n}$ is a permutation of n elements.

The third line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le n$) — if you walk into $(1, i)$ facing bottom, you should exit the grid from (n, c_i). It is guaranteed that $c_{1..n}$ is a permutation of n elements.

Output

If it is impossible to satisfy the rule, print the only number -1.

Otherwise the first line should contain a single integer m ($0 \le m \le \frac{n^2}{2}$) — the number of portals you set.

In the following m lines, each line should contain four integers x_1, y_1, x_2, y_2, represents that you set a portal consisting of two doors in (x_1, y_1) and (x_2, y_2).

If there are multiple answers, print any. You do **not** have to minimize m.

Examples

input
```
3
1 3 2
3 1 2
```
output
```
2
1 1 1 3
```

```
2 2 3 1
```

input

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

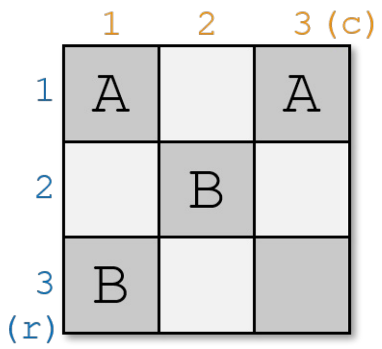
output

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

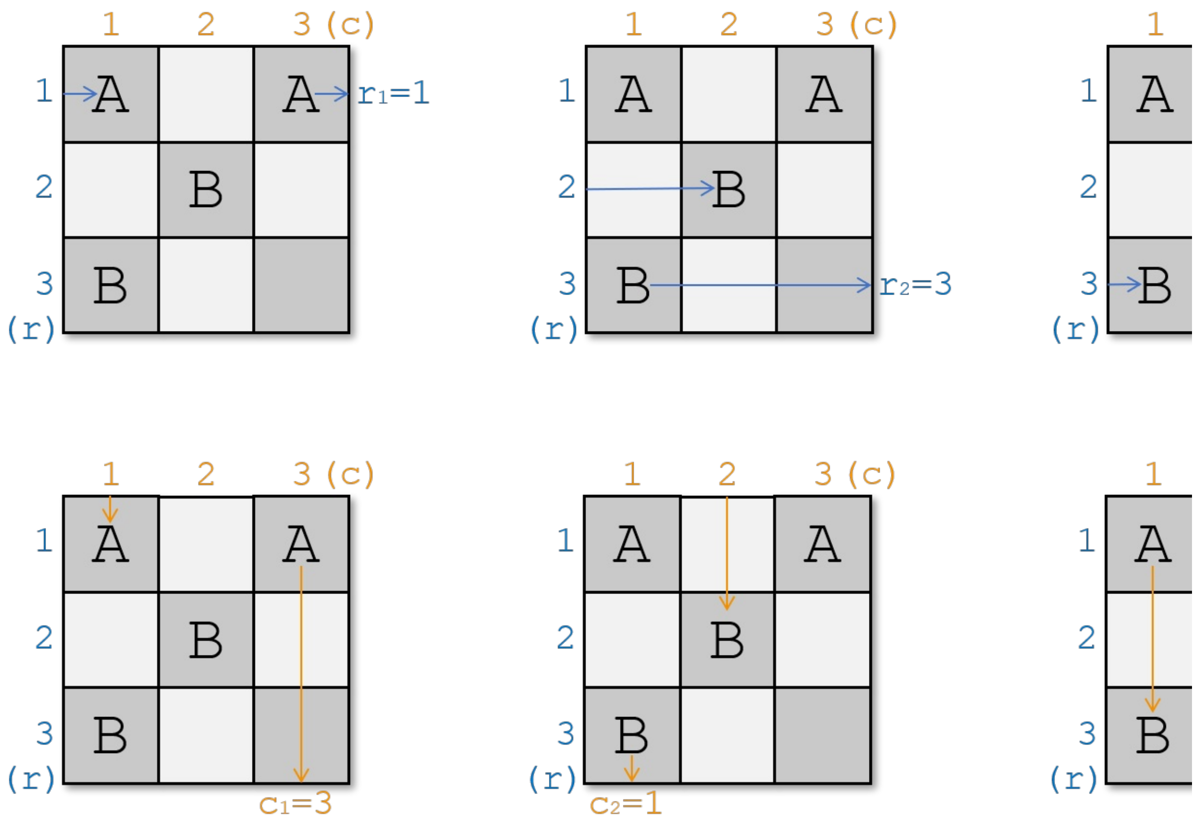
Note

Example 1

The cells with the same letter are a portal. You can set portals in this way:

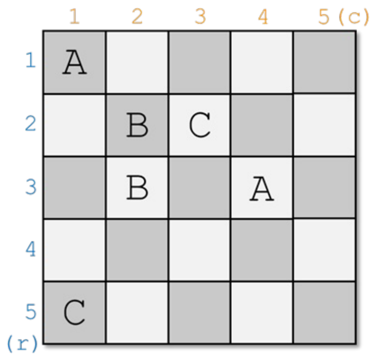


It satisfies the rule, because:



Example 2

You can set portals in this way:



1 2 3 4 5 (c)
1 2 3 4 5 (r)
A
B C
B A
C