2024 Canadian Computing Olympiad
Day 1, Problem 1
Treasure Hunt

Time Limit: 4 seconds

Problem Description

Perry the Pirate is sailing the seven seas! He has a map consisting of N islands connected by a network of M sea routes. The i-th sea route connects islands a_i and b_i and costs c_i coins to traverse in either direction. As it turns out, fighting off sea monsters can be quite expensive. In search of his next big plunder, Perry has scouted out each of the N islands and has determined that the i-th island contains a treasure chest with v_i coins inside.

It remains for him to plan out his next journey. He decides that he will sail through some (possibly empty) path of sea routes starting at island x and ending at island y. At the end of his journey, he will open the chest at island y and collect his well-earned booty.

There is one small problem though: Perry doesn't know what island he's currently on! Thus, for every possible starting island x, he would like to know the maximum possible number of coins he can earn out of all journeys starting at island x. Can you help him compute these values? You may assume Perry has enough coins to traverse any path of sea routes he chooses; he only cares about the net profit of his next journey.

Input Specification

The first line of input contains two space-separated integers N and M.

The second line of input contains N space-separated integers $v_1, v_2, \ldots, v_N$ ($0 \leq v_i \leq 10^9$).

The next M lines each contain three space-separated integers a_i, b_i ($1 \leq a_i, b_i \leq N$), and c_i ($0 \leq c_i \leq 10^9$).

It is guaranteed that there is at most one sea route between any pair of islands and each sea route connects two distinct islands.

Marks Awarded	**Bounds on N**	**Bounds on M**	**Additional constraints**
5 marks	$1 \leq N \leq 3\,000$	$0 \leq M \leq 3\,000$	None
5 marks	$1 \leq N \leq 10^6$	$0 \leq M \leq 10^6$	For all i, either $c_i = 0$ or $c_i = 10^9$
7 marks	$1 \leq N \leq 10^6$	$0 \leq M \leq 10^6$	Exactly one path of sea routes between any pair of islands
8 marks	$1 \leq N \leq 10^6$	$0 \leq M \leq 10^6$	None

Output Specification

Output N lines, where the x-th line contains the maximum possible net profit (in coins) of any journey starting at island x.

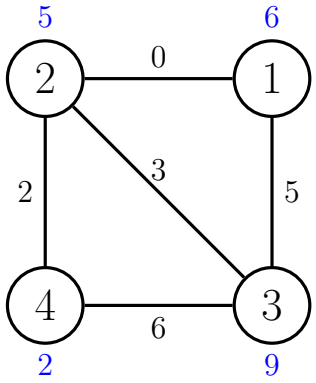
Sample Input

```
4 5
6 5 9 2
1 2 0
3 2 3
4 3 6
1 3 5
2 4 2
```

Output for Sample Input

```
6
6
9
4
```

Explanation of Output for Sample Input



For the first and third islands, it is best to just stay and open the chest on the island itself.

For the second island, Perry can travel to the first island and open the chest there. This has a net profit of $6 - 0 = 6$ coins and is the best possible net profit.

For the fourth island, Perry can to travel to the second and then the third island and open the chest there. This has a net profit of $9 - 2 - 3 = 4$ coins and is the best possible net profit.

2024 Canadian Computing Olympiad
Day 1, Problem 2
Pizza Party

Time Limit: 4 seconds

Problem Description

Troy is fleshing out the details of his latest initiative, HackCCO! Everyone knows that the biggest appeal of any hackathon is the free food. As such, to ensure the unparalleled success of HackCCO, Troy ordered a comically large cart stacked with N pizzas where the i-th pizza from the top of the cart has flavour a_i.

After the pizza cart arrives, Troy needs to arrange them into some number of stacks on a long table. To do this, he takes the pizzas one-by-one from the top of the cart and moves them onto the table, each time either placing the pizza on top of another stack of pizzas or forming a new stack on the table.

The N hungry HackCCO participants are lined up to get pizza from the table, one-by-one. Troy knows that the i-th person in line has a favourite pizza flavour of b_i. When the i-th person walks up to the table, if they see any pizzas of their favourite flavour at the top of any stack they will take any one of them at random. Otherwise, they won't take anything and will leave the table hungry.

Of course, hungry coders are not happy coders, so Troy doesn't want anyone to leave the table hungry. Thus, he asks you to help him find an arrangement of pizzas on the table such that it is possible for nobody to leave hungry. Furthermore, out of all such arrangements, Troy wants you to find one that creates the smallest number of stacks on the table (after all, tables can only get so long). Help him find such an arrangement or determine that it's impossible!

Input Specification

The first line of input contains a single integer N.

The second line of input contains N space-separated integers $a_1, \ldots, a_N$ ($1 \leq a_i \leq N$).

The third line of input contains N space-separated integers $b_1, \ldots, b_N$ ($1 \leq b_i \leq N$).

Marks Awarded	**Bounds on N**	**Additional constraints**
3 marks	$1 \leq N \leq 10^6$	$1 \leq a_i, b_i \leq 2$
4 marks	$1 \leq N \leq 5\,000$	All a_i are distinct
5 marks	$1 \leq N \leq 10^6$	All a_i are distinct
6 marks	$1 \leq N \leq 5\,000$	None
7 marks	$1 \leq N \leq 10^6$	None

[Post-CCO edit: Subtasks 4 and 5 may not be solvable efficiently. CCO competitors were judged only on subtasks 1-3.]

Output Specification

If it is impossible to arrange the pizzas as desired, output `-1`.

Otherwise, your output should consist of three lines. On the first line output K, the minimum number of stacks required. On the second line output N space-separated integers $c_1, \ldots, c_N$ ($1 \leq c_i \leq K$), indicating that the i-th pizza should be placed on stack c_i. On the third line output N space-separated integers $d_1, \ldots, d_N$ ($1 \leq d_i \leq K$), indicating that the i-th person in line takes their pizza from the d_i-th stack. This stack must have a pizza of flavour b_i at the top when the i-th participant walks up to get their pizza.

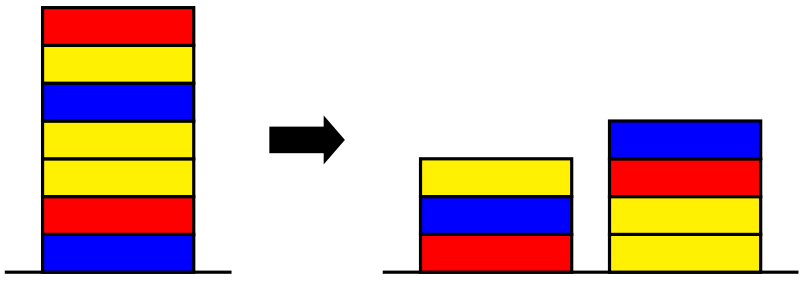
Sample Input 1

```
7
1 2 3 2 2 1 3
2 3 1 2 3 2 1
```

Output for Sample Input 1

```
2
1 2 1 2 1 2 2
1 2 2 2 1 2 1
```

Explanation of Output for Sample Input 1



An illustration of the arrangement of pizzas on the table is shown above where red, yellow, and blue boxes represent pizzas of flavours 1, 2, and 3 respectively.

Sample Input 2

```
2
1 2
1 1
```

Output for Sample Input 2

```
-1
```

2024 Canadian Computing Olympiad
Day 1, Problem 3
Summer Driving

Time Limit: 6 seconds

Problem Description

In Ontario, there are N cities numbered from 1 to N. There are $N-1$ roads numbered from 1 to $N-1$, where the i-th road connects city u_i and city v_i. It is possible to travel from any city to any other city using these roads.

Alice and Bob are travelling together, starting at city R. To make their driving experience more interesting, they devise the following game.

Alice and Bob will alternate turns, starting with Alice. On Alice's turn, she must drive along **exactly** A distinct roads that they have never traversed before in either direction. On Bob's turn, he must drive along **up to** B distinct roads (possibly zero), but some of these roads may have been traversed before.

Eventually, it will be Alice's turn, but it will be impossible for her to drive along exactly A distinct roads that they have never used before. When this happens, the game enters a final phase before Alice does any more driving. In this final phase, Bob drives along **up to** B distinct roads (possibly zero) that they have **never traversed before** in either direction.

Alice wants to end up in a city with as large a number as possible, while Bob wants to end up in a city with a small number. What is the city that Alice and Bob end their journey in when they both play optimally?

Input Specification

The first line of input contains four space-separated integers, N, R, A, and B ($1 \leq R, A, B \leq N$).

The next $N-1$ lines of input each contain two space-separated integers u_i and v_i ($1 \leq u_i, v_i \leq N, u_i \neq v_i$), describing a road.

Marks Awarded	Bounds on N	Additional Constraints
1 mark	$2 \leq N \leq 300\,000$	$A \leq B$
4 marks	$2 \leq N \leq 300\,000$	There are at most two roads connected to any city, and at most one road connected to city R.
5 marks	$2 \leq N \leq 300$	No additional constraints.
3 marks	$2 \leq N \leq 3\,000$	No additional constraints.
4 marks	$2 \leq N \leq 100\,000$	$B \leq 10$
6 marks	$2 \leq N \leq 100\,000$	No additional constraints.
2 marks	$2 \leq N \leq 300\,000$	No additional constraints.

Output Specification
Output the city that Alice and Bob end their journey in, assuming they both play optimally.

Sample Input 1
```
9 6 2 1
1 3
1 6
2 4
2 5
2 7
3 9
4 6
4 8
```

Output for Sample Input 1
```
2
```

Explanation of Output for Sample Input 1
On Alice's first turn, she has three options: Drive to city 2, 3, or 8.

If Alice drives to city 2, Bob can choose not to drive on any roads in his turn. Then, it will be impossible for Alice to drive along 2 distinct roads that were never traversed before, so the game enters the final phase. Bob can choose not to drive on any roads during this final phase, ending in city 2.

If Alice drives to city 3, Bob can choose to drive 1 road to city 1. Then, it will be impossible for Alice to drive along 2 distinct roads that were never traversed before, so the game enters the final phase. Bob's only option is to not drive on any roads during this final phase, ending in city 1.

If Alice drives to city 8, Bob can choose to drive 1 road to city 4. Then, Alice can drive to either city 5 or 7. In both cases, Bob can then drive 1 road to city 2. Alice can no longer drive along 2 distinct roads that were never traversed before, so the game enters the final phase. Bob can choose not to drive on any roads during this final phase, ending in city 2.

In all cases, Bob can force the game to end in city 1 or 2. Bob cannot force the game to end in city 1, so under optimal play, the game ends in city 2.

Sample Input 2
```
7 2 3 2
2 7
7 3
3 1
1 4
```

```
4 5
5 6
```

Output for Sample Input 2

```
3
```

2024 Canadian Computing Olympiad
Day 2, Problem 1
Infiltration

Time Limit: 1 second

Problem Description

Ondrej and Edward are spies and they are going to take down the evil organization AQT. To do so, they will need to infiltrate into the AQT base. The base can be modelled as a tree with $N = 100$ rooms, labelled from 0 to $N - 1$. Ondrej and Edward's plan to infiltrate the base is to first get kidnapped and then meet up together before executing their plan. When kidnapped, the two will be placed into different rooms unknown to each other. Once they are placed into the rooms, they will both break free at midnight and try to meet up with each other before executing their plan.

Their plan to meet up is as follows. At every odd minute, Ondrej can choose to stay at his current room or move to an adjacent room. At every even minute, Edward can choose to stay at his current room or move to an adjacent room.

A strategy is defined as the following. Let $V(A, R, T)$ denote the room agent A should be at assuming that they were at room R at midnight and it is currently T minutes after midnight. The strategy should match the conditions above. The agents are said to meet up at time $t(o, e)$, which is the first time where $V(\text{Ondrej}, o, t(o, e)) = V(\text{Edward}, e, t(o, e))$.

Ondrej and Edward want to meet up as fast as possible, relative to the distance between their two starting rooms. The distance $d(o, e)$ is the minimum number of corridors that must be traversed to reach o from e. Please help find a strategy that minimizes the maximum $\frac{t(o,e)}{d(o,e)}$ across all pairs of different rooms o and e.

Input Specification

The first line of input will contain N ($N = 100$). [Post-CCO edit: If the value of N is anything other than 100, exit the program immediately.]

The next $N - 1$ lines will each contain two space-separated integers, denoting the labels of two rooms with a bidirectional corridor between them.

Output Specification

First output a positive number T, the number of entries per starting room. Note that $T \leq 1440$ must be satisfied, otherwise you will be awarded no points.

Then, output Ondrej's strategy, followed by Edward's strategy.

To output an agent's strategy, output N lines, where the n-th line (starting from 0) represents the agent's path if they start at room n. For each line, output T spaced integers: The room

label that the agent should be in at time $1, 2, \ldots, T$.

Scoring

If the output is invalid or there exists a test case and a pair of different rooms o and e where the agents do not meet at or before time T, then no points will be awarded.

Otherwise, let S be the maximum among all test cases and pairs of o and e ($o \neq e$) of the value of $\frac{t(o,e)}{d(o,e)}$. The following table shows how the available 25 marks are distributed:

Score	**Bounds on S**
3	$200 < S \leq 1440$
6	$100 < S \leq 200$
8	$50 < S \leq 100$
10	$40 < S \leq 50$
12	$30 < S \leq 40$
15	$25 < S \leq 30$
17	$20 < S \leq 25$
18	$19 < S \leq 20$
19	$18 < S \leq 19$
20	$17 < S \leq 18$
21	$16 < S \leq 17$
22	$15 < S \leq 16$
25	$S \leq 15$

Sample Input 1

```
5
0 2
3 2
1 4
2 4
```

Output for Sample Input 1

```
8
2 2 4 4 1 1 1 1
1 1 1 1 1 1 1 1
3 3 2 2 3 3 2 2
3 3 2 2 0 0 2 2
4 4 4 4 2 2 2 2
0 2 2 3 3 3 3 2
```

```
1 4 4 2 2 0 0 0
2 3 3 3 3 3 3 3
3 3 3 3 3 3 3 3
4 1 1 1 1 4 4 4
```

Explanation of Output for Sample Input 1

Note that this is an invalid test case as $N \neq 100$, so it will not appear in the test cases when judging. The output for the test case is valid. Note that this would not score any points because if Ondrej starts at room 1 and Edward starts at room 3, then they will never meet each other.

2024 Canadian Computing Olympiad
Day 2, Problem 2
Heavy Light Decomposition

Time Limit: 4 seconds

Problem Description
In an array containing only positive integers, we say an integer is heavy if it appears more than once in the array, and light otherwise.

An array is good if the integers in the array alternate between light and heavy.

Given an array $a_1, \ldots, a_N$, count the number of ways to partition it into some number of contiguous subarrays such that each subarray, when considered as an array on its own, is good. As the answer may be large, output it modulo 1 000 003.

Input Specification
The first line of input contains a single integer, N.

The next line contains N integers $a_1, \ldots, a_N (1 \leq a_i \leq N)$.

Marks Awarded	Bounds on N	Additional Constraints
3 marks	$2 \leq N \leq 50\,000$	For each i, $a_i \leq 26$.
4 marks	$2 \leq N \leq 5\,000$	No additional constraints.
5 marks	$2 \leq N \leq 500\,000$	If i is odd, then $a_i = 1$.
6 marks	$2 \leq N \leq 500\,000$	Any number appears at most twice in the array.
7 marks	$2 \leq N \leq 500\,000$	No additional constraints.

Output Specification
The number of ways to partition the array into good contiguous subarrays, modulo 1 000 003.

Sample Input 1
```
5
1 2 3 2 3
```

Output for Sample Input 1
```
4
```

Explanation of Output for Sample Input 1
There are four valid partitions of [1, 2, 3, 2, 3]:

- [1], [2], [3], [2], [3]
- [1], [2, 3, 2], [3]
- [1], [2], [3, 2, 3]
- [1, 2, 3, 2], [3]

Sample Input 2

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

Output for Sample Input 2

```
6
```

2024 Canadian Computing Olympiad
Day 2, Problem 3
Telephone Plans

Time Limit: 4 seconds

Problem Description

The "Dormi's Fone Service" is now the only telephone service provider in CCOland. There are N houses in CCOland, numbered from 1 to N. Each telephone line connects two distinct houses such that all the telephone lines that ever exist form a forest.

There is an issue where the phone lines are faulty, and each phone line only exists for a single interval of time. Two houses can call each other at a certain time if there is a path of phone lines that starts at one of the houses and ends in the other house at that time.

We would like to process Q queries of the following forms:

- `1 x y`: Add a phone line between houses x and y. It is guaranteed that a phone line between houses x and y was never added before.
- `2 x y`: Remove the phone line between houses x and y. It is guaranteed that a phone line currently exists between houses x and y.
- `3 t`: Compute the number of pairs of different houses that can call each other at some time between the current query and t queries ago. To be more clear, let G_q be the state of CCOland after the q-th query, where G_0 is the state of CCOland before any queries. If this is the s-th query, then count the number of pairs of houses that are connected in at least one of $G_{s-t}, G_{s-t+1}, \ldots, G_s$.

Also, some test cases may be encrypted. For the test cases that are encrypted, the arguments `x`, `y`, or `t` are given as the bitwise xor of the true argument and the answer to the last query of type `3` (if there have been no queries of type `3`, then the arguments are unchanged).

Input Specification

The first line of input will contain E ($E \in \{0, 1\}$). $E = 0$ denotes that the input is not encrypted, while $E = 1$ denotes that the input is encrypted.

The second line contains two space-separated integers N and Q, representing the number of houses in CCOland and the number of queries, respectively.

The next Q lines contain queries as specified above (queries are encrypted or not depending on E).

For the q-th query ($1 \leq q \leq N$), it is guaranteed that (after decrypting if $E = 1$) $1 \leq x, y \leq N$ for type `1` and `2` queries and $0 \leq t \leq q$ for type `3` queries.

Marks Awarded	Bounds on N	Bounds on Q	Encrypted?
3 marks	$1 \leq N \leq 30$	$1 \leq Q \leq 150$	$E = 0$
2 marks	$1 \leq N \leq 30$	$1 \leq Q \leq 150$	$E = 1$
4 marks	$1 \leq N \leq 2\,000$	$1 \leq Q \leq 6\,000$	$E = 0$
2 marks	$1 \leq N \leq 2\,000$	$1 \leq Q \leq 6\,000$	$E = 1$
4 marks	$1 \leq N \leq 100\,000$	$1 \leq Q \leq 300\,000$	$E = 0$
5 marks	$1 \leq N \leq 100\,000$	$1 \leq Q \leq 300\,000$	$E = 1$
5 marks	$1 \leq N \leq 500\,000$	$1 \leq Q \leq 1\,500\,000$	$E = 1$

Output Specification

For each query of type `3`, output the answer to the query on a new line.

Sample Input 1

```
0
4 12
3 0
1 1 2
3 0
1 1 3
3 0
3 5
2 2 1
3 0
3 8
1 1 4
3 0
3 11
```

Output for Sample Input 1

```
0
1
3
3
1
3
3
5
```

Explanation of Output for Sample Input 1

This test case is not encrypted.

For the 1st query, no pairs of different houses could have called each other.

For the 3rd query, only houses 1 and 2 could have called each other.

For the 5th query, $\{(1,2),(1,3),(2,3)\}$ is the set of pairs that could have called each other. The 6th query is the same.

For the 8th query, only houses 1 and 3 could have called each other.

For the 9th query, there is a point in time where $\{(1,2),(1,3),(2,3)\}$ could have called each other.

For the 11th query, the set of pairs that could have called each other is $\{(1,3),(1,4),(3,4)\}$.

For the 12th query, the set of pairs that could have called each other at any previous time is $\{(1,2),(1,3),(1,4),(2,3),(3,4)\}$.

Sample Input 2

```
1
4 12
3 0
1 1 2
3 0
1 0 2
3 1
3 6
2 1 2
3 3
3 9
1 2 7
3 3
3 8
```

Output for Sample Input 2

```
0
1
3
3
1
3
3
5
```

Explanation of Output for Sample Input 2

Encrypted version of sample 1.