PROBLEM 4

Human LCD II

Time and memory limits: 1 second, 256 MB

With your previous performance met with widespread critical acclaim, you've been hired once again to work full-time to make informatics competitions entertaining. While trapeze artists swing wildly from the scaffolding and a marching band parades around the contestants, you'll be taking centre stage with a new routine of the giant human LCD.

You've managed to arrange some volunteers in a grid of R rows and C columns, and have provided each of them with glowsticks of **three** distinct colours. These colours are called "0", "1", and "2".[1] At the moment, each of the volunteers holds up exactly one of their three glowsticks; specifically, the volunteer in the ith row and jth column displays the colour A_{ij}.

The organisers have given you a target R by C display — with B_{ij} denoting the colour of the ith row and jth column — which they want you to form with the human LCD. To do this, you can issue *instructions*: instructing a particular person causes them to switch their displayed colour to any colour of your choice (one of "0", "1", or "2").

However, there is one problem. In order to keep the display from getting too monotonous, the organisers mandate that no two adjacent people (directly to the left, right, front, or back) can ever hold up the same colour at the same time.

Your job is to issue some sequence of these instructions to transform the display A into the display B. For 60% of the points, this can be any sequence with length at most one million. For full score, however, you must find a routine with minimum length.

Subtasks and Constraints

Your program will be graded using many secret tests. Every test follows some rules:

- $1 \leq R, C \leq 1000$.
- $R \times C \leq 1000$.
- $0 \leq A_{ij}, B_{ij} \leq 2$ for all i and j.
- In both A and B, no two adjacent cells have the same colour.

The secret tests are divided into subtasks. Your score for each subtask is calculated as described in the Scoring section.

- For Subtask 1 (20 points), $R = 1$ and $C = 3$.
- For Subtask 2 (20 points), $R = 1$, $A_{1i} = (i-1) \pmod 3$ and $B_{1i} = (10^9 - i) \pmod 3$ for all i.[2]
- For Subtask 3 (10 points), $R = 1$, $A_{1i} = (i-1) \pmod 3$ and $B_{1i} = (C - i) \pmod 3$ for all i.
- For Subtask 4 (30 points), $R = 1$.
- For Subtask 5 (20 points), no special rules apply.

[1] The event ran out of the necessary budget to buy rights to use "red", "green", and "blue".
[2] In other words, $A_1 = [0, 1, 2, 0, 1, 2, \ldots]$ and $B_1 = [0, 2, 1, 0, 2, 1, \ldots]$.

Input

Your solution must read input and print output. We recommend using the solution templates (which you can find on the competition website) to help you with input and output.

The input follows a specific format:

- The first line contains the integers R and C.
- The next R lines describe the initial arrangement A.
  - The ith line contains C integers representing A_{ij} for $1 \leq j \leq C$.
- The next R lines describe the target arrangement B.
  - The ith line contains C integers representing B_{ij} for $1 \leq j \leq C$.

Output

- The first line must contain a single nonnegative integer L representing the number of instructions used.
  - For full points, this should be the minimum possible.
- The next L lines should each contain 3 integers X_i, Y_i, and V_i.
  - The ith line represents an instruction that switches the colour displayed in the X_ith row and Y_ith column to V_i.

Scoring

For each test case:

- If your output contains more than one million instructions, contains an invalid instruction, causes two adjacent people to display the same colour at any point, or does not transform A into B, you will receive 0%.
- Otherwise, if your sequence uses the minimum possible number of instructions, you will receive 100%.
- Otherwise, you will receive 60%.

Your score for each subtask is the minimum percentage scored across all test cases in that subtask, multiplied by the number of points allocated to that subtask.

Sample Input 1

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

Sample Output 1

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

Sample Input 2

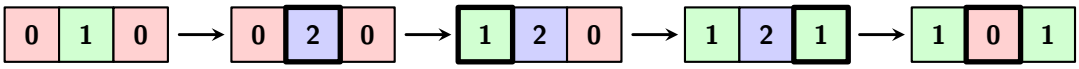
```
2 3
0 1 2
1 2 0
1 2 0
2 0 1
```

Sample Output 2

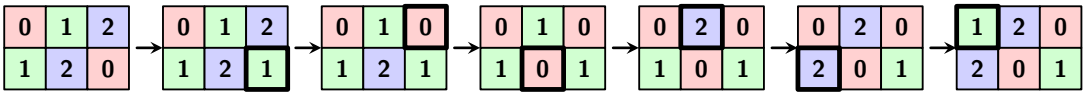
```
6
2 3 1
1 3 0
2 2 0
1 2 2
2 1 2
1 1 1
```

Explanation

In the 1st sample case, we can switch to a temporary colour to dodge what would otherwise be a set of adjacent equal colours:



In the 2nd sample case, one way of getting to the target grid is as follows:



Note that other possible instruction sequences exist.