Problem B

Eternity II

The Eternity II puzzle[1] was released in 2007 with the promise to pay $2 million to the first person to present a complete solution. However, up until today no correct solution was presented and the prize remains unclaimed.

Eternity II is a classic edge-matching puzzle which involves placing 256 square tiles into a 16×16 grid. Each tile has its edges marked with different shape/color combinations (which we will simply call color here). The tiles must be placed in such a way that all the colors on their edges precisely match the colors of the adjacent tiles. The borders of the grid are a special case and match only tiles with gray edges. Tiles can be rotated; therefore, each tile has 4 possible placements for each grid position. There are 22 colors, not including the gray edges. On the original puzzle, the center tile is pre-determined and some tile positioning hints are given.

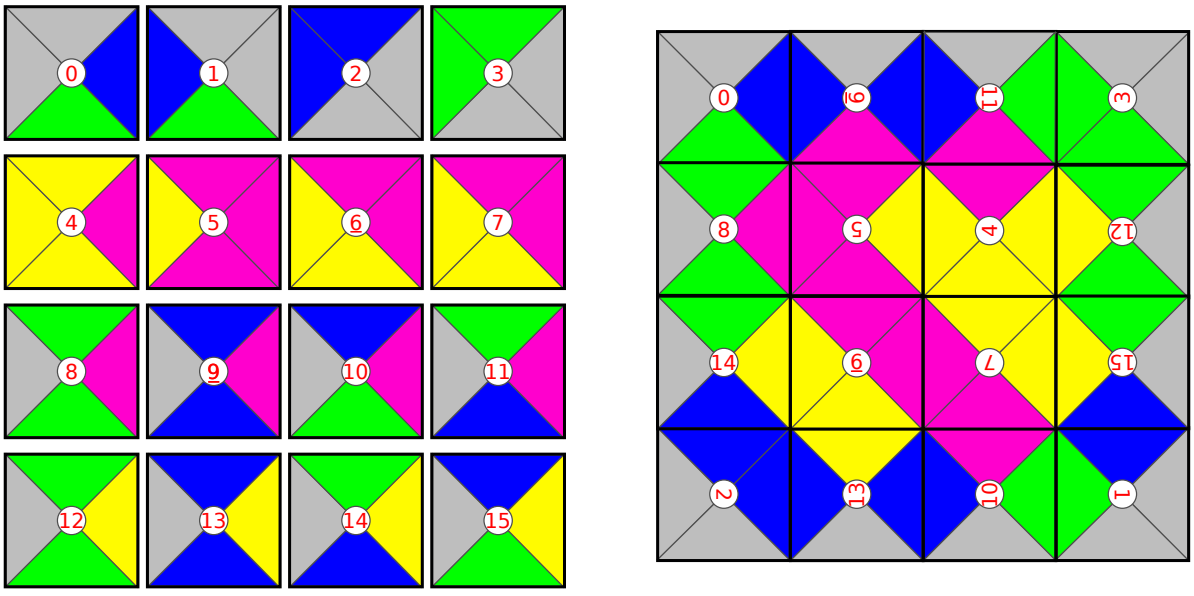

Figure B1. On the left, a set of tiles for a 4×4 puzzle. On the right the solved puzzle. Notice how this solution rotates some of the tiles.

This puzzle was designed to be difficult to solve by brute-force computer search, and remains intractable on its original configuration. Indeed, the number of possible configurations

[1] The game description was adapted from https://en.wikipedia.org/wiki/Eternity_II_puzzle

(assuming all the pieces are distinct, and ignoring the fixed pieces and tile positioning hints) is 256! × 4^{256} roughly \$1.15 × 10^{661}. A tighter upper bound can be obtained taking into account the fixed tile in the center and the positioning hints yielding a search space of 3.11 × 10^{545}.

Since this competition must end before the end of the universe, here we will deal which much smaller instances of the same problem. However, to keep things interesting we will not provide any hints or tiles with predefined positions. The puzzle grid size, number of colors and tiles will be given through the standard input and the solution should be presented using the standard output.

You were given a sequential version of a solver which uses a naïve brute-force backtracking method. Your task is to write a parallel version of this code. Feel free to use any heuristic or method to improve the performance of the sequential version. Notice, however, that it might be the case that a single input has multiple distinct correct solutions. This will not be a problem as long as the solution provided by your code is correct since the automated evaluation system already takes this into consideration.

Input

Each input contains one puzzle. It consists of a list of integers separated by spaces and new lines. The first line contains 2 integers: the grid size g and the number of colors c. The next g^2 lines list the tiles. The order of the tiles is important (it will be used for the output) and is counted from 0, thus tiles are numbered from 0 to $g^2 - 1$. Each tile is given by 4 integers between 0 and $c - 1$ describing the colors of its edges in clockwise order, starting from the top edge. The color 0 (gray) is considered to be a special case: the only acceptable color for the borders. The example input below represents the input tiles shown in Figure B1(a) You may assume $g \leq 16$ and $c \leq g$.

The input must be read from the standard input.

Output

The expected output must have g^2 lines, each one representing one of the cells of the grid. The order of the lines follows the grid from left to right, top to bottom. Each line is composed by 2 integers, the first indicates the tile number, and the second the number of clockwise

rotations needed for that tile. The expected output corresponding to the solution presented in Figure B1(b) is shown in the next section.

The output must be written to the standard output.

Example

Input	Output for the input
4 5 0 1 2 0 0 0 2 1 1 0 0 1 2 0 0 2 3 4 3 3 4 4 4 3 4 4 3 3 4 4 3 3 2 4 2 0 1 4 1 0 1 4 2 0 2 4 1 0 2 3 2 0 1 3 1 0 2 3 1 0 1 3 2 0	0 0 9 1 11 1 3 3 8 0 5 2 4 3 12 2 14 0 6 0 7 2 15 2 2 1 13 3 10 3 1 1