Scramble Squares Puzzle Solver

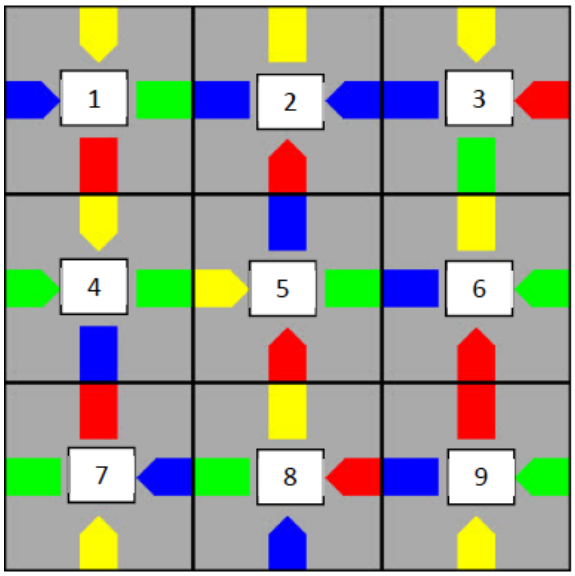
1. Description:

The object of the Scramble Squares puzzle game is to arrange the nine colorfully illustrated square tiles into a large square so that the graphics on the tiles' edges match perfectly to form a completed design in every direction.

The 9-piece puzzle creates an enormous number of possible combinations in a 3-piece x 3-piece pattern.

$9! \, x \, 4^9 = 95,126,814,720$, to be exact

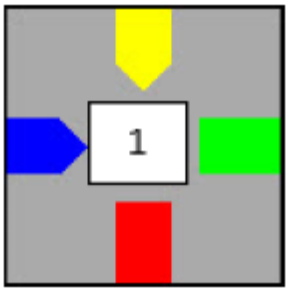
For most of these puzzles, there is one solution, but sometimes several unique solutions do exist. Take, for instance, the puzzle below:



Reading from left to right, top to bottom, we can encode the pieces as follows:

```
Y1,G0,R0,B1
Y0,B1,R1,B0
Y1,R1,G0,B0
Y1,G0,B0,G1
B0,G0,R1,Y1
Y0,G1,R1,B0
R0,B1,Y1,G0
Y0,R1,B1,G0
R0,G1,Y1,B0
```

Each line of the input file describes a tile in the scrambled image. The tiles are numbered, and the images on each edge are encoded as four 2-character strings, each separated by a comma. In this puzzle, Y stands for yellow, G for green, R for red, and B for blue, but any upper-case alphabetic character could have been used. 0 is for the “tail” half of the image, which in this case is the body of the arrow. 1 is for the “head” part, which is the point of the arrow. The images on the tiles’ edges are described clockwise, starting at top. For example, in tile 1,



the yellow arrow-head comes first, followed by the green arrow body, red arrow body, and blue arrow head. That orientation is encoded by:

```
Y1,G0,R0,B1
```

Note: The input file will be valid. You do not have to worry catching errors in the input file.

The only legal operations are tile rotations and swaps, and the puzzle is only considered complete when the same color heads and tails match up for all tiles.

2. Output:

Your goal is to write a program to find ALL unique solutions to an arbitrary puzzle. Solutions that are merely rotations of one another should be discarded. The solution with the lowest tile ordering should be kept. For example, if “123456789” happens to be a solution, so would “741852963”, “987654321”, and “369258147”. Keep only “123456789”.

```
$ ./puzzlesolver puzzles/arrows.txt
Input tiles:
1. <Y1, G0, R0, B1>
2. <Y0, B1, R1, B0>
3. <Y1, R1, G0, B0>
4. <Y1, G0, B0, G1>
5. <B0, G0, R1, Y1>
6. <Y0, G1, R1, B0>
7. <R0, B1, Y1, G0>
8. <Y0, R1, B1, G0>
9. <R0, G1, Y1, B0>

2 unique solutions found:
+--------+--------+--------+
|1  G0   |2  B0   |4  G0   |
|Y1   R0 |R1   Y0 |Y1   B0 |
|   B1   |   B1   |   G1   |
+--------+--------+--------+
|9  B0   |6  B0   |7  G0   |
|Y1   R0 |R1   Y0 |Y1   R0 |
|   G1   |   G1   |   B1   |
+--------+--------+--------+
|3  G0   |8  G0   |5  B0   |
|R1   B0 |B1   Y0 |Y1   G0 |
|   Y1   |   R1   |   R1   |
+--------+--------+--------+

+--------+--------+--------+
|3  R1   |9  Y1   |7  Y1   |
|Y1   G0 |G1   B0 |B1   G0 |
|   B0   |   R0   |   R0   |
+--------+--------+--------+
|8  B1   |6  R1   |2  R1   |
|R1   G0 |G1   B0 |B1   B0 |
|   Y0   |   Y0   |   Y0   |
+--------+--------+--------+
|5  Y1   |1  Y1   |4  Y1   |
|R1   B0 |B1   G0 |G1   G0 |
|   G0   |   R0   |   B0   |
+--------+--------+--------+
```

The output must include the original input, a blank line, the number of unique solutions, and then the solutions drawn in ASCII art showing the correct rotation (with a blank line between each solution). Basically, you'd like your program to show someone exactly how to solve a puzzle. For the puzzle above, your output must match that on the last page of this document. The puzzles should be displayed in lexicographic order if the tile numbers were arranged left-to-right, top-to-bottom in a single string.

Of course, if there are no solutions, your program should output "No solution found." after listing the input tiles. Furthermore, since we want to be grammatically correct, if there is only 1 unique solution, make sure your program outputs "1 unique solution found:" In another words, make the word solution singular.

3. Execution time:

The program must run as fast as possible, no solution should take more than 2ms, use timing functions to check your work

4. Language:

C, Java are permitted. Java developers should name the file PuzzleSolver.java, while in c, the file should be named puzzlesolver.cpp