Challenge 8 - Tuenti Restructuration

« Prev Next »

Tuenti, which is well known as an awesome social network, decided to change its business model and create a mobile network company called Tuenti Móvil. Since all engineers don't have the same technical knowledge, the company needs to move people around and put those who have different knowledge close together so they may learn from each other, benefit from peer programming, and even have romantic dinners.

The TechLead (TL) has prepared the new table layout but he doesn't want his next project to be delayed by moving all the engineers at the same time (they tend to get distracted and have Nerf® gun fights if everyone moves at the same time).
His plan is to swap two adjacent people (*) every day until the new table layout is complete. A table layout can be represented by a 3x3 matrix. Obviously, position 2, 2 is not available since it's a hole, but it can be used as a swapping movement.

Your work is to help the TL by ascertaining the minimum number of days needed to move all the engineers around the table. If the new table layout is impossible to execute, your algorithm should return -1.

(*) Two persons are adjacent if they share the same horizontal/vertical edge.

Input

The first line (t) contains the number of tables that need to be reorganised. Then 2t tables follow.
Each table consists of a 3x3 matrix describing the existing table layout with the name (Na) of each engineer, followed by a 3x3 matrix describing the target table layout.

The input data for successive tables are separated by a blank line.
Constraint:

- 1<=t<=10000
- Only UTF-8 characters
- 1<=len(Na)<=20

Output

For each table print a single line containing the shortest number days needed to get the new table layout. If there is no way to reach the target state, print the number -1.

Example

Sample input

```
1

Javier, Andrew, Vincent
Marcio, , Bartek
Einar, Goran, Ignacio

Andrew, Javier, Vincent
Marcio, , Bartek
Einar, Goran, Ignacio
```

Sample output

```
1
```

Submit & test your code

To test and submit code we provide a set of tools to help you. Download contest tools if you haven't already done that. You will then be able to test your solution to this challenge with the challenge tokens.

```
Challenge tokens: CHALLENGE_8, CHALLENGE_SUBMIT_8
```

To test your program

```
./test_challenge CHALLENGE_8 path/program
```

A nice output will tell you if your program got the right solution or not. You can try as many times as you need.

To test your program against the input provided in the submit phase

```
./test_challenge CHALLENGE_SUBMIT_8 path/program
```

During the submit phase, in some problems, we might give your program harder inputs. As with the test token, a nice output will tell you if your program got the right solution or not. You can try as many times as you need.

In the actual contest you first need to solve the test phase before submitting the code, you must provide the source code used to solve the challenge and you can only submit once (once your solution is submitted you won't be able to amend it to fix issues or make it faster).

If you have any doubts, please check the info section.

« Prev Next »