Challenge 8 - Tuenti Restructuration

Tuenti, well know as an awesome social network, decided to change its business model and create a mobile network company (known as Tuenti Móvil). Because all engineers don’t have the same technical knowledge, the company needs to move people around and put people with different knowledge close together so they can learn from each other, do peer programming and even have a romantic dinner if they want to!

The TechLead (TL) did the new table plan but doesn’t want to have his next project delayed by moving all engineers at the same time (they used to do get distracted and do Nerf® guns fights when moving together).
His plan is to swap two adjacent persons(*) every day until the new table plan is complete. A table plan can be represented by a 3x3 matrix. Of course the position 2,2 is not available (it’s a hole) but it can be used as a swapping movement.

Your work is to help the TL by knowing the minimum number of days needed to move all engineers around the table. If the new table plan is impossible to do, 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 a current table plan with the name (Na) of each engineer, followed by a 3x3 matrix describing the goal table plan.

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 organisation of the table. If there is no way to reach the final 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 and submit your solution to this challenge with the challenge token.

```
Challenge token: kGvhxfGa6hp7YtzLRbwS
```

To test your program

```
./test_challenge kGvhxfGa6hp7YtzLRbwS 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 submit your program to the challenge

```
./submit_challenge kGvhxfGa6hp7YtzLRbwS
path/source_pkg.tgz path/program
```

Note that you first need to solve the test phase before submitting the code. During the submit phase, in some problems, we might give your program harder questions, so try to make your program failsafe.

Important: In this phase, you must provide the source code used to solve the challenge and, if necessary, a brief explanation of how you solved it.

Remember **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, so please be sure your solution is finished before submitting it.

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

Go ahead

I'm done! :)

Once you have submitted your code, hit refresh and continue to next challenge.

I'm stuck! :(

Be sure you follow the Tuenti Engineering twitter for updates and possible hints during the contest.

If this challenge is too hard and you are blocked, you will be able to skip it after two hours. Note that **you won't be able to complete it later**, and you have a limited number of challenges to skip.

Finally, if you run out of skips but are still really stuck with one problem, you will be able to skip it after 24 hours.

Challenge status:

Test case	Not done
Solution submitted	Not done
Skip	You still have to wait 0h, 30m and 0s to be able to skip this challenge

Refresh status