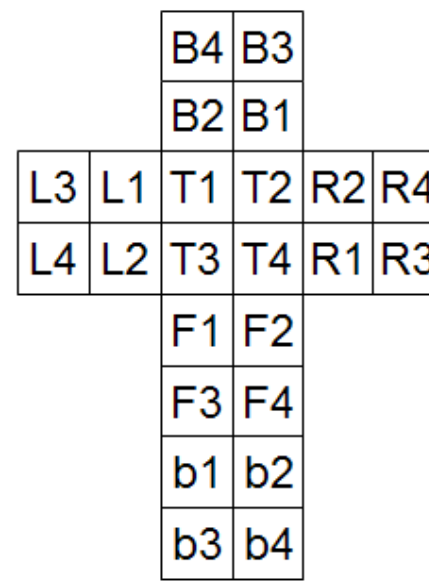
Everyone knows about the Rubik's cube, even though few people can actually solve it. For a computer this is no match of course, but finding the optimal solution (least number of turns) is a very hard task which requires a lot of computational resources.

In this problem we (or rather, your solution) will solve a smaller cube instead: the $2 \times 2 \times 2$ mini cube (see picture). But don't let the size fool you. Most people won't be able to solve this cube either!

Given the cubes initial configuration, determine the least number of quarter turns (one side turned 90 degrees) required to bring the cube in order. You can assume that the initial state is valid so a solution will always exist. When the cube is solved it will look like in the picture to the left, with white ('`W`') on the opposite side of blue ('`B`'), red ('`R`') on the opposite side of orange ('`O`'), and yellow ('`Y`') on the opposite side of green ('`G`').

Input

The first line in the input will contain the number of cubes your program should solve (at most 100). Each cube will then be described in 6 lines, one for each face on the cube in the following order: top (T), front (F), right (R), bottom (b), back(B) and left (L). Each face will described with four characters indicating their colors, in the order given by the folded out cube to the right (please excuse the somewhat strange-looking order ...) A blank line will preceed each input case.



Output

For each cube, output on a line by itself the number of quarter turns required to solve the cube.

Sample Input

```
2

GGGG
WWBB
OORR
YYYY
BBWW
RROO

WRRG
WORO
BGBW
YYOG
BORY
GYWB
```

Sample Output

```
2
6
```