Problem A
Rush Hour Puzzle

Time limit: 2 seconds
Memory limit: 1024 megabytes

Problem Description

Rush Hour is a puzzle game invented by Nob Yoshigahara in the 1970s. It is now being manufactured by **ThinkFun**. The board is a 6×6 grid with grooves in the tiles to allow vehicles to slide. Cars and trucks are both one square wide, but cars are two squares long and trucks are three squares long. Vehicles can only be moved forward or backward along a straight line on the grid. The goal of the game is to get the only red car totally out through the exit of the board by moving the other vehicles out of its way. Figure 1 gives an example of Rush Hour puzzle.



Figure 1: An example of Rush Hour puzzle.

We give each vehicle of a puzzle a unique *id*, numbered from 1 to the number of vehicles, in which the red car's id is 1. The board information of a puzzle is represented by a 6×6 matrix, named *board matrix*. Each entry of a board matrix is the id of the vehicle placed on that groove, and the entries are filled with 0 if there exists no vehicle on those grooves. The exit of the board is located at the right end side of the 3rd row. Figure 2 shows the board matrix corresponding to the puzzle in Figure 1.

Moving a piece (car or truck) by one unit (a groove) is called a *step*. A puzzle is *easy* if it can be solved (the red car totally out through the exit of the board) in no more than 10 steps. Please write a program to judge whether a puzzle is easy or not.

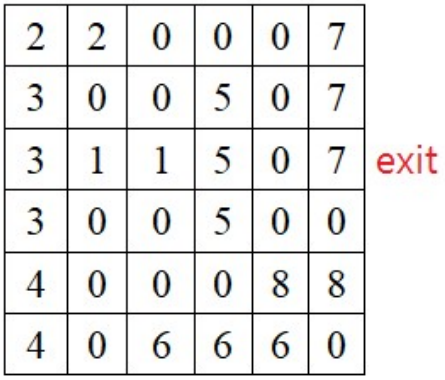


Figure 2: The board matrix corresponding to the puzzle in Figure 1.

Input Format

The input contains 6 lines, each line indicates the content (6 integers separated by a blank) of each row of a board matrix.

Output Format

Output the minimum number of steps for solving the input puzzle if the puzzle is easy, otherwise output `-1`.

Technical Specification

- There are at most 10 vehicles on the board for each puzzle.
- Only the red car can be moved out of the board for each puzzle.

Sample Input 1

```
2 2 0 0 0 7
3 0 0 5 0 7
3 1 1 5 0 7
3 0 0 5 0 0
4 0 0 0 8 8
4 0 6 6 6 0
```

Sample Output 1

```
-1
```

Sample Input 2

```
0 2 0 6 6 0
0 2 0 0 7 0
0 3 1 1 7 0
0 3 4 4 8 0
0 5 5 5 8 0
0 0 0 0 0 0
```

Sample Output 2

```
6
```