

HOME TOP CATALOG CONTESTS GYM PROBLEMSET GROUPS RATING EDU API CALENDAR HELP RAYAN 🖫

PROBLEMS SUBMIT STATUS STANDINGS CUSTOM TEST

A. Beautiful Matrix

time limit per test: 2 seconds memory limit per test: 256 megabytes

You've got a 5×5 matrix, consisting of 24 zeroes and a single number one. Let's index the matrix rows by numbers from 1 to 5 from top to bottom, let's index the matrix columns by numbers from 1 to 5 from left to right. In one move, you are allowed to apply one of the two following transformations to the matrix:

- 1. Swap two neighboring matrix rows, that is, rows with indexes i and i+1 for some integer i $(1 \le i \le 5)$.
- 2. Swap two neighboring matrix columns, that is, columns with indexes j and j+1 for some integer j ($1 \le j \le 5$).

You think that a matrix looks *beautiful*, if the single number one of the matrix is located in its middle (in the cell that is on the intersection of the third row and the third column). Count the minimum number of moves needed to make the matrix beautiful.

Input

The input consists of five lines, each line contains five integers: the j-th integer in the i-th line of the input represents the element of the matrix that is located on the intersection of the i-th row and the j-th column. It is guaranteed that the matrix consists of 24 zeroes and a single number one.

Output

Print a single integer — the minimum number of moves needed to make the matrix beautiful.

Examples



→ Attention

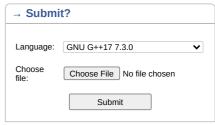
The package for this problem was not updated by the problem writer or Codeforces administration after we've upgraded the judging servers. To adjust the time limit constraint, a solution execution time will be multiplied by 2. For example, if your solution works for 400 ms on judging servers, then the value 800 ms will be displayed and used to determine the verdict.

Codeforces Round 161 (Div. 2) Finished Practice



Start virtual contest





→ Problem tags	
implementation *800	
	No tag edit access

→ Contest materials	
Announcement	×
• Tutorial	×