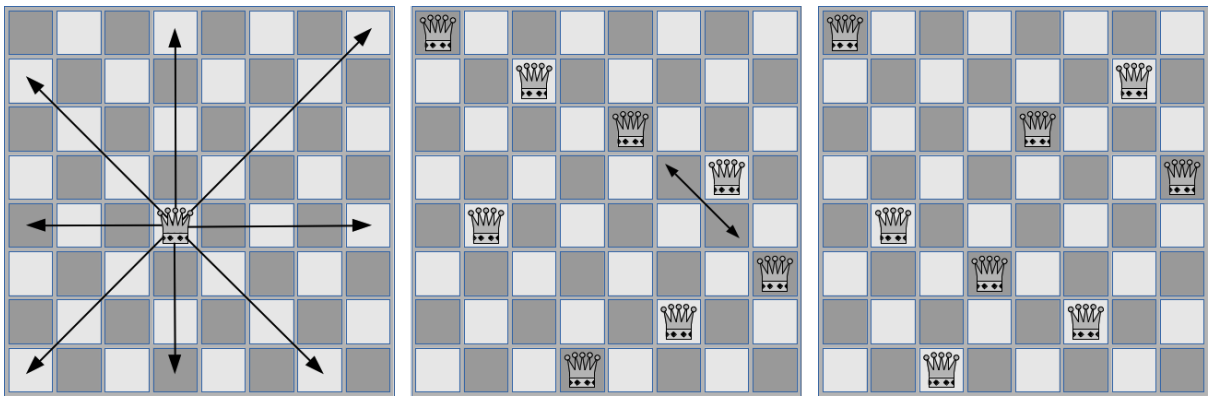
Ob-CHESS-ion

The Association of Creative Minds (ACM) is organizing the International Creative Puzzles Competition (ICPC) for enthusiastic puzzle-solvers. This competition, usually called ACM-ICPC, is going to be held next week in Singapore, particularly in NUS School of Computing. They are searching for a talented NUS student to help them organize this competition. After interviewing many candidates, they have decided that you are the most suitable candidate out of all applicants. Congratulations!

Now they are going to brief you about your responsibilities in this competition. Out of the many puzzles being contested, one is a very famous puzzle called the N-queens puzzle. It is a puzzle in which you have to distribute N queens on an N x N chessboard such that no two queens can attack each other. Their goal is to find someone who can solve this puzzle first, and award that person with a huge sum of money. However, one problem arises: they want to verify whether the submitted solution is correct or not. A manual check is possible, albeit slow (not to mention tedious). They don't want participants to complain about the slowness of the committee in checking the solutions, do they?

Being a beginner at chess, you don't know what a queen is or how it moves. Luckily, the ACM-ICPC organizer is very helpful. They give you this diagram to illustrate the nature of the puzzle:



1. A queen's movement is defined in the left diagram: i.e. a queen can step along any number of squares at each move, along one of the directions illustrated by the arrows in the left diagram.
2. A queen can "attack" another queen if there is another queen that is reachable in a single move. If you look at the middle diagram, the two queens connected by the arrow can attack each other, hence it is not a valid solution to the N-queens puzzle, given N = 8.
3. The right diagram illustrates a valid solution to the N-queens puzzle, given N = 8.

You have one job. Help verify whether the submitted solutions are correct. More precisely, write an automated checker to check whether the given input is a valid solution to the N-queens puzzle. Good luck!

Input

The input consists of a single number **N** (4 <= N <= 200). **N** rows follow. Each row represents the i^{th} row of the chess board. In each row, there are N numbers, separated by a single space, with 0 denoting an empty cell and 1 denoting a cell with a queen on it. There are **precisely N queens** distributed in the chessboard.

Output

If the given input is a valid solution to the N-queens puzzle, print "VALID". Otherwise, print "INVALID".

Sample Input 1

```
8
1 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 1
0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0
0 0 1 0 0 0 0 0
```

Sample Output 1

VALID

Sample Input 2

```
8
1 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0
0 0 0 0 1 0 0 0
0 0 0 0 0 0 1 0
0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 1
0 0 0 0 0 1 0 0
0 0 0 1 0 0 0 0
```

Sample Output 2

INVALID

Explanation

Sample input 1 is illustrated in the right diagram above and sample input 2 is illustrated in the middle diagram. It is clear that sample input 1 provides a valid solution since no two queens can attack each other, whereas in sample input 2, as shown in the middle diagram, there is a pair of queens that can attack each other, hence it is an invalid solution.

Skeleton

You are given the skeleton file Queens.java

```
/*
 * Name        :
 * Matric No.         :
 * Plab Account       :
 */

import java.util.*;

public class Queens {

    public static void main(String[] args) {

    }

}
```

Notes:

1. You should develop your program in the subdirectory ex1 and use the skeleton java file provided. You should not create a new file or rename the file provided.
2. If your algorithm is different from the given skeleton, you are free to write a solution according to your own algorithm.
3. You do not have to use OOP for this sit-in lab.
4. You are free to define your own methods.
5. Please be reminded that the marking scheme is:

Input	: 10%	**Correctness**	: 50%
Output	: 10%	**Programming Style**	: 30%