Pandachess

Mr. Panda is trying to solve the *rook problem* on an infinitely large chessboard. The *rook problem* is to place rooks on a chessboard so that no 2 rooks lie on the **same row** or the **same column**. Mr. Panda has tried to come up with a solution and shows you his solution. However, you realise it is wrong! This saddens Mr. Panda. In an attempt to cheer him up, you want to find out what is the number of ways you can **remove at least one rook** from his solution to get a valid solution to the *rook problem*. The valid solution must **contain at least one rook.**

Input
The input contains an integer **N** on the first line. This will be followed by **N** lines with 2 integers on each line representing the row and column of a rook respectively. It is guaranteed that no 2 rooks are on the same square.

In addition, it is also guaranteed that there are at least 2 rooks that are in the same row or column. In other words, the original positions of the rooks are **not** a valid solution to the rook problem.

Output
The output should contain a single integer, representing the number of ways you can remove some number of rooks from his solution to get a valid solution to the *rook problem*.

Limits

- $2 \le \mathbf{N} \le 12$
- All row and column numbers will be between 0 and 10^9 inclusive.

Sample Input (**pandachess1.in**)	Sample Output (**pandachess1.out**)

`3`
`10 19`
`2 8`
`10 8`
	`4`

Explanation
There are 6 ways to remove at least 1 rook such that at least 1 rook is left:

Removed Rook(s)	**Rook(s) Left**	**Is it valid?**
(10, 19)	(2, 8), (10, 8)	No, the 2 rooks share the same column 8.
(2, 8)	(10,19), (10, 8)	No, the 2 rooks share the same row 10.
(10, 8)	(10, 19), (2, 8)	Yes
(10, 19), (2, 8)	(10, 8)	Yes
(10, 19), (10, 8)	(2, 8)	Yes
(2, 8), (10, 8)	(10, 19)	Yes

Out of the 6 ways, only 4 of them leave a valid solution to the rook problem so the answer is 4.

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. You are free to define your own helper methods and classes (or remove existing ones).
3. Please be reminded that the marking scheme is:
   a. Public Test Cases (1%) - 1% for passing **all** test cases, 0% otherwise
   b. Hidden Test Cases (1%) - Partial scoring depending on test cases passed
   c. Manual Grading (1%)
      i. Overall Correctness (correctness of algorithm, severity of bugs)
      ii. Coding Style (meaningful comments, modularity, proper indentation, meaningful method and variable names)
4. Your program will be tested with a time limit of not less than **2 sec** on Codecrunch.

Skeleton File – Pandachess.java

You are given the below skeleton file `Pandachess.java`. You should see a non-empty file when you open the skeleton file. Otherwise, you might be in the wrong working directory.

```java
/**
 * Name          :
 * Matric. No    :
 * PLab Acct.    :
 */

import java.util.*;

public class Pandachess {
    private void run() {
        //implement your "main" method here
    }

    public static void main(String[] args) {
        Pandachess newPandachess= new Pandachess();
        newPandachess.run();
    }
}
```