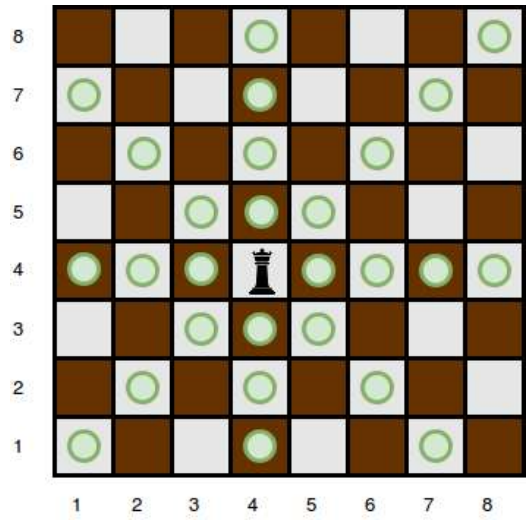
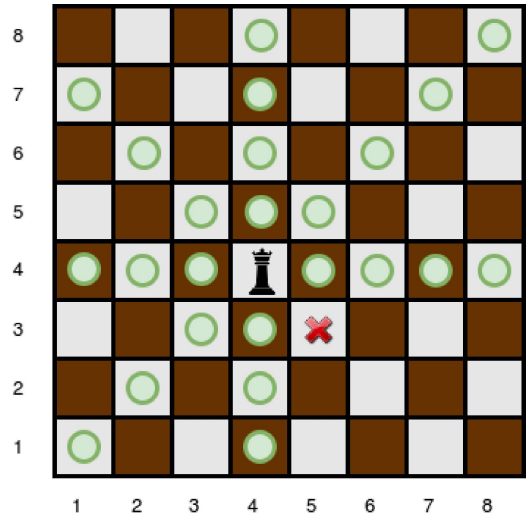
You will be given a square chess board with one queen and a number of obstacles placed on it. Determine how many squares the queen can attack.

A queen is standing on an $n \times n$ chessboard. The chess board's rows are numbered from 1 to n, going from bottom to top. Its columns are numbered from 1 to n, going from left to right. Each square is referenced by a tuple, (r, c), describing the row, r, and column, c, where the square is located.

The queen is standing at position (r_q, c_q). In a single move, she can attack any square in any of the eight directions (left, right, up, down, and the four diagonals). In the diagram below, the green circles denote all the cells the queen can attack from $(4, 4)$:



There are obstacles on the chessboard, each preventing the queen from attacking any square beyond it on that path. For example, an obstacle at location $(3, 5)$ in the diagram above prevents the queen from attacking cells $(3, 5)$, $(2, 6)$, and $(1, 7)$:



Given the queen's position and the locations of all the obstacles, find and print the number of squares the queen can attack from her position at (r_q, c_q). In the board above, there are 24 such squares.

Function Description

Complete the queensAttack function in the editor below.

```
}

string ltrim(const string &str) {
    string s(str);
    s.erase(s.begin(), find_if(s.begin(), s.end(), no
    (isspace))));
    return s;
}

string rtrim(const string &str) {
    string s(str);
    s.erase(find_if(s.rbegin(), s.rend(), not1(ptr_fu
    s.end());
    return s;
}

vector<string> split(const string &str) {
    vector<string> tokens;
    string::size_type start = 0, end = 0;
    while ((end = str.find(" ", start)) != string::np
        tokens.push_back(str.substr(start, end - star
        start = end + 1;
    }
    tokens.push_back(str.substr(start));
    return tokens;
}
```

Line: 99 Col: 2

Upload Code as File | Run Code | Submit Code

Test against custom input

You have earned 30.00 points!

You are now 70 points away from the 2nd star for your problem solving badge.

0% 30/100

Problem Solving

Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

Test case 0
Test case 1
Test case 2
Test case 3
Test case 4
Test case 5

Compiler Message

Success

Input (stdin) Download

```
4 0
4 4
```

Expected Output Download

```
9
```