HackerRank | Prepare > Algorithms > Implementation > Queen's Attack II

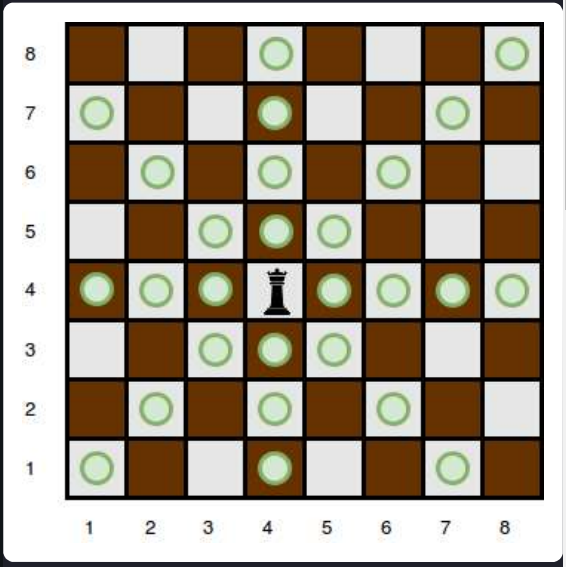
Exit Full Screen View

Problem | Submissions | Leaderboard | Discussions | Editorial

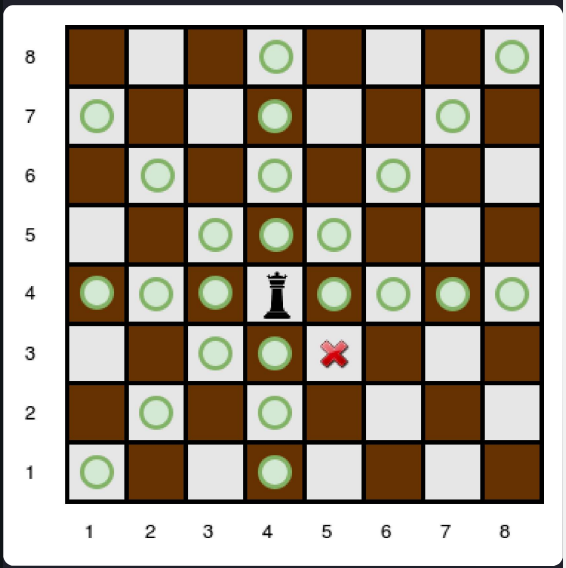
You will be given a square chess board with one queen and a number of obstacles placed on it. Determine how many squares the queen can attack.

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.

Change Theme Language C++20

```
726 int queensAttack(int n, int k, int r_q, int c_q, vector
27              r += dir.first;
28              c += dir.second;
29
30              // Stop if out of board or obstacle encount
31              if (r < 1 || r > n || c < 1 || c > n || obs
                ({r, c})) {
32                  break;
33              }
34
35              moves++;
36          }
37      }
38
39      return moves;
40  }
41
42  int main() {
43      int n, k, r_q, c_q;
44      cin >> n >> k;
```

Line: 55 Col: 1

Upload Code as File | Test against custom input | Run Code | Submit Code

Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

- Sample Test case 0
- Sample Test case 1
- Sample Test case 2

Input (stdin) Download

```
4 0
4 4
```

Your Output (stdout)

```
9
```

Expected Output Download

```
9
```