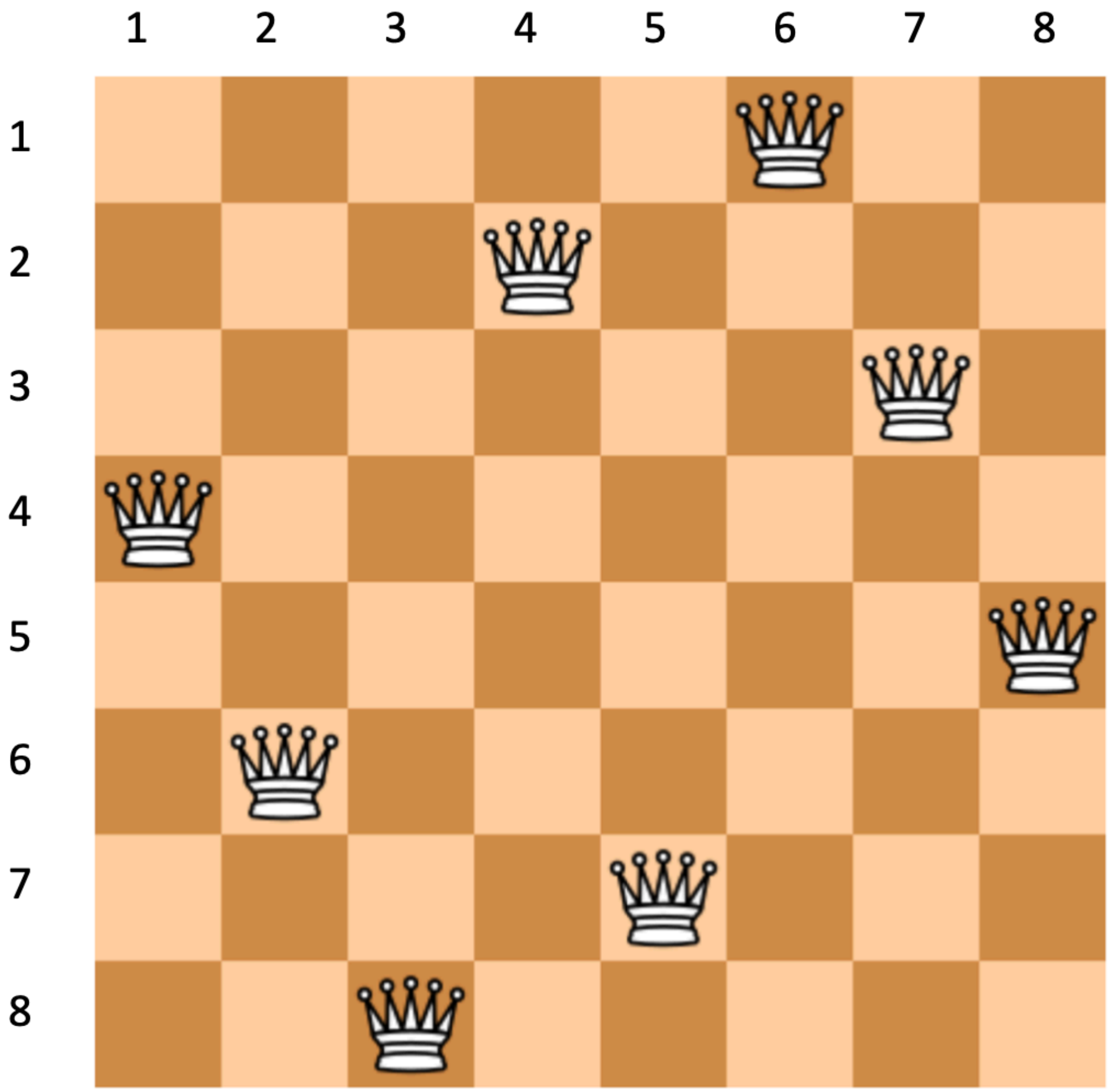
Assg#2 The K-Queen Problem

due: 12/14 (at 11:59pm)

Write a program that finds all possible ways to place K queens on an K×K chessboard so that no two queens threaten each other. Thus, a solution requires that no two queens share the same row, column, or diagonal, as shown in the diagram below:



Let (1,1) be the top-left corner and x, y be the vertical as well as horizontal axes, respectively. We represent this solution as

`(1,4), (2,6), (3,8), (4,2), (5,7), (6,1), (7,3), (8,5)`. For simplicity, since the x-axis is counted according to numerical order, we can simply represent this solution as the following 8 numbers: `4 6 8 4 7 1 3 5`.

What you need to do

The input will be the number of `K`. All you need to do is output **ALL SOLUTIONS** in **lexicographic order**. If we count different rotations and flippings as different solutions, then there are 92 solutions for case K=8. You should output these solutions in lexicographic order (e.g. 15863724 should be placed before 17468253). **Your program must be able to handle at least the case** `K=8`. Your program should have reasonable efficiency. You can use what you have learned so far to write this assignment: both C or C++ will be fine. Since this assignment does not involve complex data types, you probably don't need to use a lot of C++ object oriented programming techniques. Most likely you need to either use a stack or recursion (which contains an implicit system stack).

Grading Criteria

Correctness 70% + Coding style 30%

You should follow the Github coding style as this contributes to 30% of your grades. Note that you will ONLY get style points if your program is correct on at least some of our hidden cases. In other words, if your program cannot work, you won't get any style points.

Make sure you pass all test cases before submission. Any submissions after the deadline will **NOT be counted**.

Submission

No need to submit a README file

Submit your sourcecode (with comments) both on OJ and eLearn.

Sample Input 1

```
4
```

Sample Output 1

```
2 4 1 3
3 1 4 2
2 solution(s) for the 4-Queen problem.
```

Sample Input 2

```
1
```

Sample Output 2

```
1
1 solution(s) for the 1-Queen problem.
```

Sample Input 3

```
8
```

Sample Output 3

```
1 5 8 6 3 7 2 4
1 6 8 3 7 4 2 5
1 7 4 6 8 2 5 3
1 7 5 8 2 4 6 3
2 4 6 8 3 1 7 5
2 5 7 1 3 8 6 4
2 5 7 4 1 8 6 3
2 6 1 7 4 8 3 5
2 6 8 3 1 4 7 5
2 7 3 6 8 5 1 4
2 7 5 8 1 4 6 3
2 8 6 1 3 5 7 4
3 1 7 5 8 2 4 6
3 5 2 8 1 7 4 6
3 5 2 8 6 4 7 1
3 5 7 1 4 2 8 6
3 5 8 4 1 7 2 6
3 6 2 5 8 1 7 4
3 6 2 7 1 4 8 5
```

```
3 6 2 7 5 1 8 4
3 6 4 1 8 5 7 2
3 6 4 2 8 5 7 1
3 6 8 1 4 7 5 2
3 6 8 1 5 7 2 4
3 6 8 2 4 1 7 5
3 7 2 8 5 1 4 6
3 7 2 8 6 4 1 5
3 8 4 7 1 6 2 5
4 1 5 8 2 7 3 6
4 1 5 8 6 3 7 2
4 2 5 8 6 1 3 7
4 2 7 3 6 8 1 5
4 2 7 3 6 8 5 1
4 2 7 5 1 8 6 3
4 2 8 5 7 1 3 6
4 2 8 6 1 3 5 7
4 6 1 5 2 8 3 7
4 6 8 2 7 1 3 5
4 6 8 3 1 7 5 2
4 7 1 8 5 2 6 3
4 7 3 8 2 5 1 6
4 7 5 2 6 1 3 8
4 7 5 3 1 6 8 2
4 8 1 3 6 2 7 5
4 8 1 5 7 2 6 3
4 8 5 3 1 7 2 6
5 1 4 6 8 2 7 3
5 1 8 4 2 7 3 6
5 1 8 6 3 7 2 4
5 2 4 6 8 3 1 7
5 2 4 7 3 8 6 1
5 2 6 1 7 4 8 3
5 2 8 1 4 7 3 6
5 3 1 6 8 2 4 7
5 3 1 7 2 8 6 4
5 3 8 4 7 1 6 2
5 7 1 3 8 6 4 2
5 7 1 4 2 8 6 3
5 7 2 4 8 1 3 6
5 7 2 6 3 1 4 8
5 7 2 6 3 1 8 4
5 7 4 1 3 8 6 2
5 8 4 1 3 6 2 7
5 8 4 1 7 2 6 3
6 1 5 2 8 3 7 4
6 2 7 1 3 5 8 4
6 2 7 1 4 8 5 3
```

```
6 3 1 7 5 8 2 4
6 3 1 8 4 2 7 5
6 3 1 8 5 2 4 7
6 3 5 7 1 4 2 8
6 3 5 8 1 4 2 7
6 3 7 2 4 8 1 5
6 3 7 2 8 5 1 4
6 3 7 4 1 8 2 5
6 4 1 5 8 2 7 3
6 4 2 8 5 7 1 3
6 4 7 1 3 5 2 8
6 4 7 1 8 2 5 3
6 8 2 4 1 7 5 3
7 1 3 8 6 4 2 5
7 2 4 1 8 5 3 6
7 2 6 3 1 4 8 5
7 3 1 6 8 5 2 4
7 3 8 2 5 1 6 4
7 4 2 5 8 1 3 6
7 4 2 8 6 1 3 5
7 5 3 1 6 8 2 4
8 2 4 1 7 5 3 6
8 2 5 3 1 7 4 6
8 3 1 6 2 5 7 4
8 4 1 3 6 2 7 5
92 solution(s) for the 8-Queen problem.
```