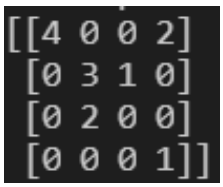
Some testcases are included in this document for easy visualisation of the outputs. All the testcases are present in the tests folder.

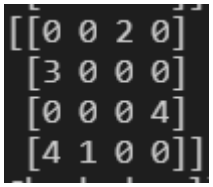
Q.1. In this question, the input consists of an integer k, and a csv file which contains two $k \times k$ sudokus. The program will a required solution and output the answer. The input need not have a unique solution, in which case the program will simply output one of the satisfying solutions.

- k = 2

Input :

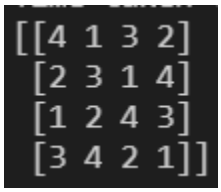


Sudoku 1

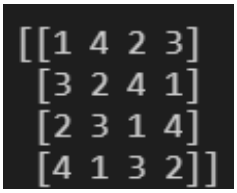


Sudoku 2

Output :



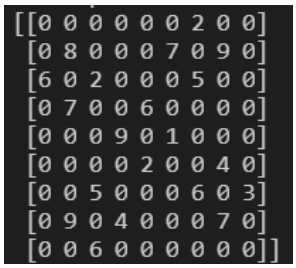
Sudoku 1



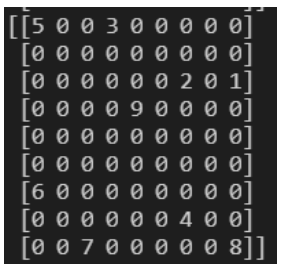
Sudoku 2

- k = 3

Input :

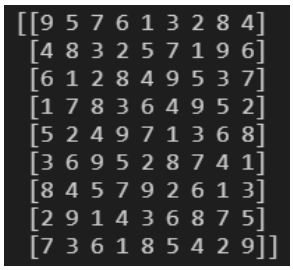


Sudoku 1

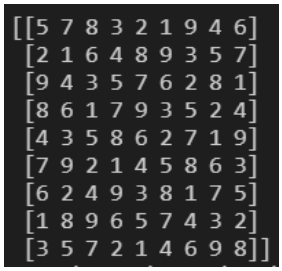


Sudoku 2

Output :



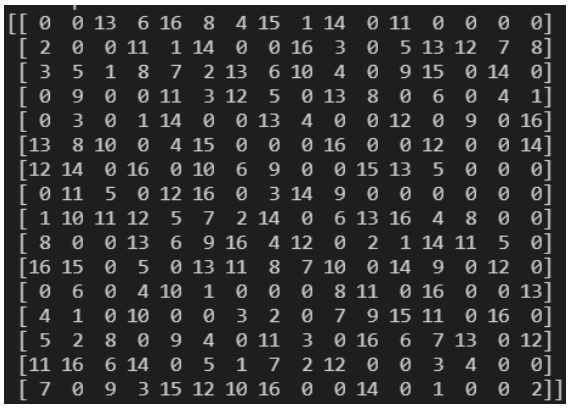
Sudoku 1



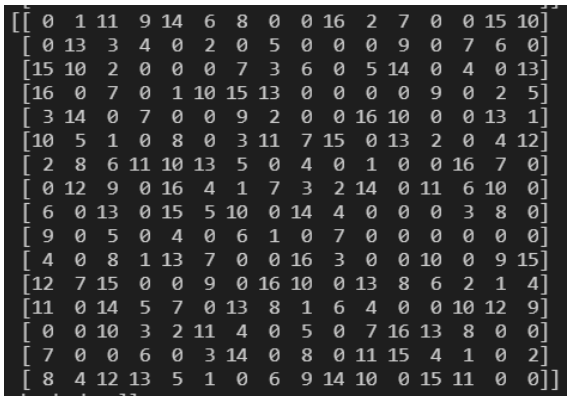
Sudoku 2

- k = 4

Input :

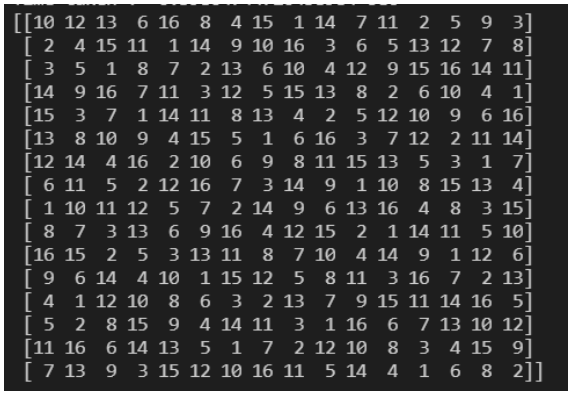


Sudoku 1

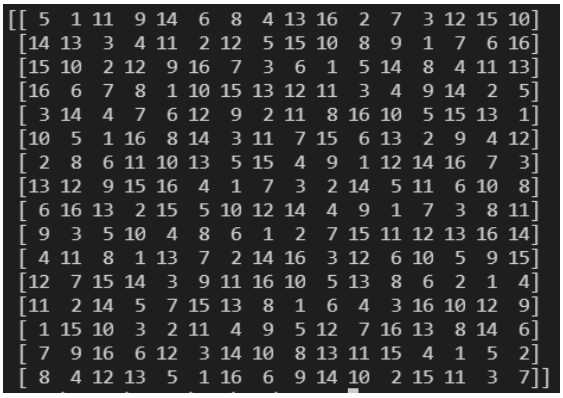


Sudoku 2

Output :



Sudoku 1



Sudoku 2

Q. 2. In this part, the input will be **k,** and the output will be **a pair of sudokus**. Since, the outputs each time will be random, the outputs might not match exactly with those shown here. Hence, we are just mentioning how the outputs will look like for three values of k. **The exact entries should differ each time we run the code.**

- k = 2

```
[[0 3 0 0]
 [4 0 0 0]
 [0 0 0 0]
 [3 0 0 2]]
```

Sudoku 1

```
[[0 0 1 0]
 [0 0 3 4]
 [1 0 0 3]
 [0 0 0 0]]
```

Sudoku 2

- k = 3

```
[[0 5 4 6 1 0 7 2 8]
 [7 2 0 0 0 0 0 0 0]
 [3 0 0 0 8 0 0 5 9]
 [0 3 2 0 7 0 0 0 0]
 [5 9 1 3 2 0 8 0 4]
 [8 0 7 0 0 5 2 0 3]
 [0 8 5 7 0 0 0 0 2]
 [0 4 0 0 9 8 6 1 0]
 [0 7 0 0 0 0 3 8 0]]
```

Sudoku 1

```
[[0 0 0 1 0 0 0 6 7]
 [1 0 0 8 2 6 9 0 4]
 [6 5 0 7 3 4 2 8 1]
 [8 6 5 0 0 0 7 0 0]
 [0 0 2 0 0 7 6 0 0]
 [0 0 4 5 6 0 8 0 0]
 [0 2 0 3 0 8 0 0 0]
 [0 0 0 0 0 1 5 7 3]
 [0 3 0 6 0 5 0 2 0]]
```

Sudoku 2

- k = 4

```
[[ 0 12 13  6  0  0  4  0  0 14  7 11  9  5  2  0]
 [ 2  4  0  0  1 14 10  0 16  0  5  0 13 12 11  7]
 [ 0  0  0  0  7  0  0  0  0  0  0  0  0 16 14  0]
 [ 0 16  0  7  0  0 12  0  0  0  0  2  0 10  0  0]
 [ 0  3  0  1  0 11  0  0  0  6  2 13  0  0  5  0]
 [ 0  0  0  4 15  1  0  0  0 12  3 16 14  6  7 11]
 [ 5  0  0 16 12 13  6  0  0  0 10  0  3  0  0 15]
 [ 6 11 12  0  9 16  7  3 14 15  0  0  0  8 13  4]
 [ 0  0  0  0  8  7  2 14  0  9 15  0  4  1  0 13]
 [ 1  0  0 15  5  0 16 12  6  2  0  0  0 11  3  0]
 [ 0 13  0 14  3  6 11  0  7  0  4  8  0  0  0  0]
 [ 8  0  0  9  0  4  0 13  5  0  0  1  0  7 12 14]
 [ 4  1  0  0  0 10  0 16 13  0  0 15  0  0  0  0]
 [ 9  2  0  3  4  0 14 11 12  0  0 10  7 13  0  0]
 [ 0 14  6  0 13  5  1  7  0  0  0  0 12  4  0  9]
 [ 7 15  0  0  0  0  0  0  0  5 14  0  1  3 10  2]]
```

Sudoku 1

```
[[ 0  1  0  0  0  6  8  4 13 16  0  0  3  0  0  0]
 [14 13  0  4  0  2  0  0  0 10  8  0  1  0  6 16]
 [ 0  0  0 12  9 16  7  3  0  1  5  0  8  0  0  0]
 [ 0  0 16  8  0  0  0  0  0  0  0  0  0  0 14  9  0]
 [10 14  4  7  6  0  0  2 11  8 16  3  0 15 13  0]
 [ 0  0  1 11  0 14  0 10  0 15  6  0  0  9  0  0]
 [ 2  8  0 13  0  0  0 11  5  9  1  0 14 10  0  3]
 [ 3 12  9 15  0 13  1  7  4  0  0  0  0 11  6  0  0]
 [ 6 15  0  2 16  5  0  0  0  0  0  1  0  0  0 11]
 [ 9  0  0 10  0  8  6  1  2  0  0  0  0 13  0  0]
 [ 4  0  0  1 13  7  0  0  0  0 12  6  9  5  0  0]
 [ 0  0 14 16  3  9 11  0  0  0  0  0  0  2  0  6]
 [11  0 10  3  0 15 13  8  0  0  0  5  6  0  0  9]
 [ 1  0 15  0 12  0  0  0  3 13  7 16  0  0  0 14]
 [13 16  0  6 10  3 14  0  0 12  0  2 15  1  5  0]
 [ 0  4  0 14  2  1  5 16  0  0  0 15  0 11  3  7]]
```

Sudoku 2