Sudoku Solver

Implementation:

I used a literal a_ijl_ which denotes that the value at the ith row and jth column is l.
It is either true or false.
For using cnf form, I used the formula x=k**4*(i-1)
+ k**2*(j-1) + l

The following 5 checks were implemented in the program for normal sudoku:

1. At least one number is present in the square
2. At most one number is present in the square
3. A number appears in the row only once
4. A number appears in the column only once
5. A number appears in the k x k square only once

These were the general rules for solving sudoku.
Then I solved sudoku 1 with these constraints and then for the additional constraints S1(i,j)!=S2(i,j), I solved the sudoku 2 with these additional constraints.

If both the sudokus are solvable with the given constraints, it will print them both. If S1 is not solvable, then it will say so and end the program. If S1 is solvable but S2 is not with the given constraints, it will print the answer to S1 and print S2 is not solvable.

Assumption:

It is assumed that the user will change the name of the CSV file according to the “Readme” file before each run and will input the value ok k. The CSV file has values according to the instructions given in the question.

Limitations:

It can be a bit slow and with higher k values like k=6, it can be a bit hard to read the answer as some numbers are single digit(1,2,3…) while others are double digits(10, 11, …). This messes with the orientation of the numbers. I tried using more indentation for proper spacing but then one single line split into two lines, so I didn’t go for it.

Sudoku Generator

Implementation:

I randomly assigned a number to a random square of the sudoku and then solved the sudoku. Then I removed each number one by one and check if more solutions exist after removing it. If only one solution exist, then we delete that number and if multiple solutions exist, then we keep the number.

To get sudoku pair, I just increased the value of each number present in sudoku 1, i.e., change 1 to 2, 2 to 3, 3 to 4, and so on.

Assumption:

It is assumed that the user will input the value of k and rename the file created so it does not get overwritten.
Also the program should be run on a machine which can handle more resources.

Limitations:

For k>=5, it takes a lot of time and a lot of memory. Not recommended: if it takes too much RAM please exit the program.
It can be a bit hard to read the answer as some numbers are single digit(1,2,3…) while others are double digits(10, 11, …). This messes with the orientation of the numbers. I tried using more indentation for proper spacing but then one single line split into two lines, so I didn't go for it.