Intermediate Submission Questions

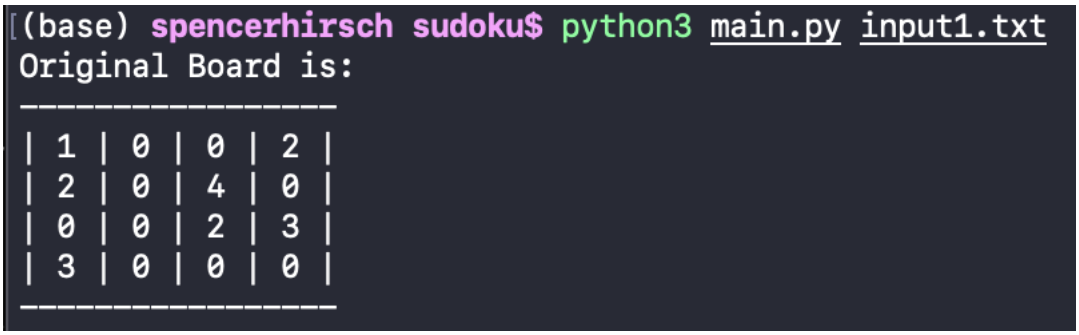
Spencer Hirsch, Thomas Johnson

February 4, 2023

Psuedo-Code for Backtracking Algorithm:

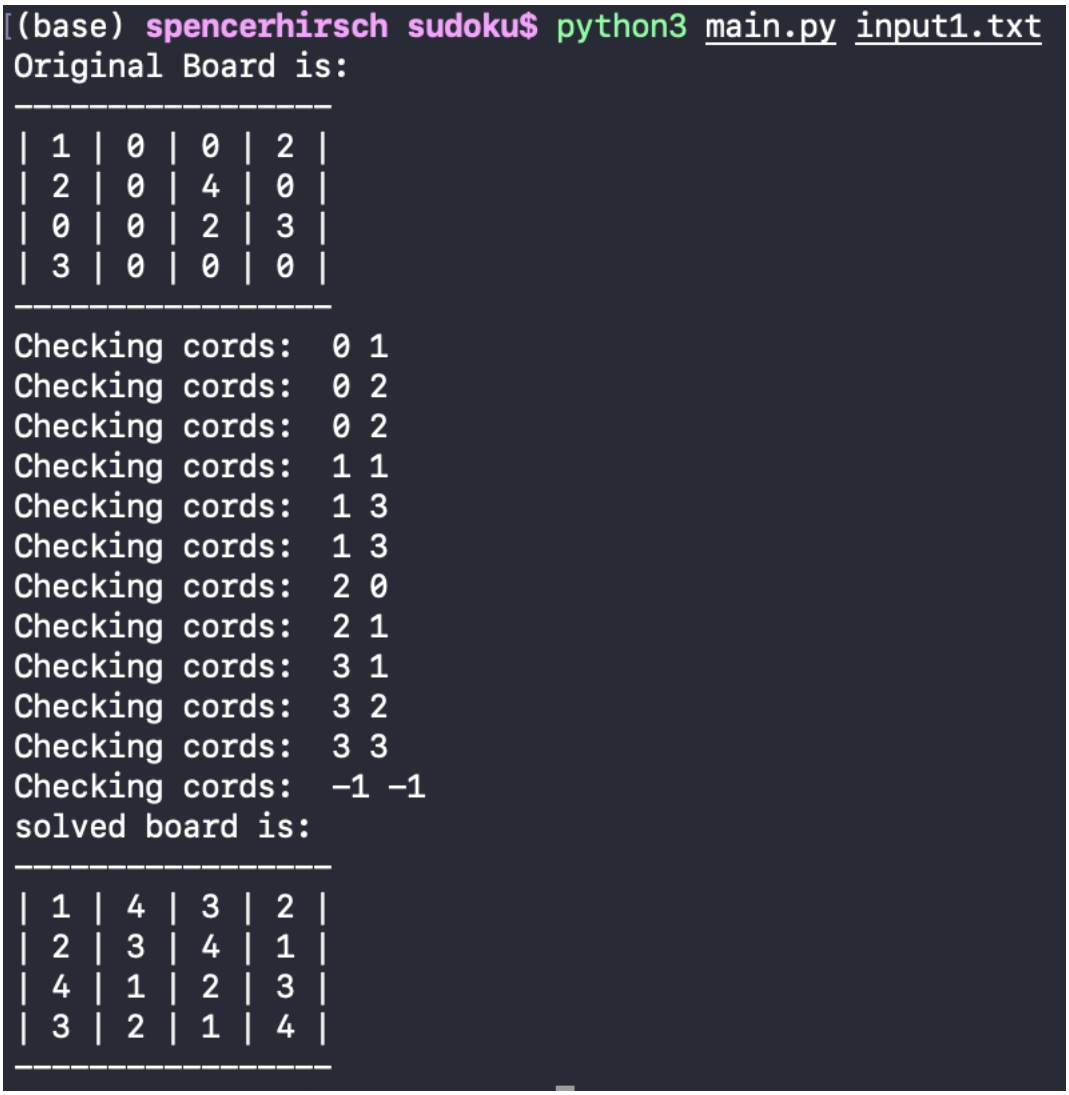
```
useful funcs to make:
valid_move_list() -> returns a list of valid numbers a cell can take
    check rows, and columns for unused numbers


solve(board) true or false
    if board if full
        save board
        return true


    for open cell in board
        nums = valid_move_list(cell)
        for each num in nums
            if cell can be filled with num
                fill it
                if (solve(board))
                    return true / filled up board with valid moves


            mark cell empty / backtracks


        return false / the open cell didnt have a valid number to put, making this
            ↪ patheway unsolveable
```

Screenshot demonstrating compilation of code:



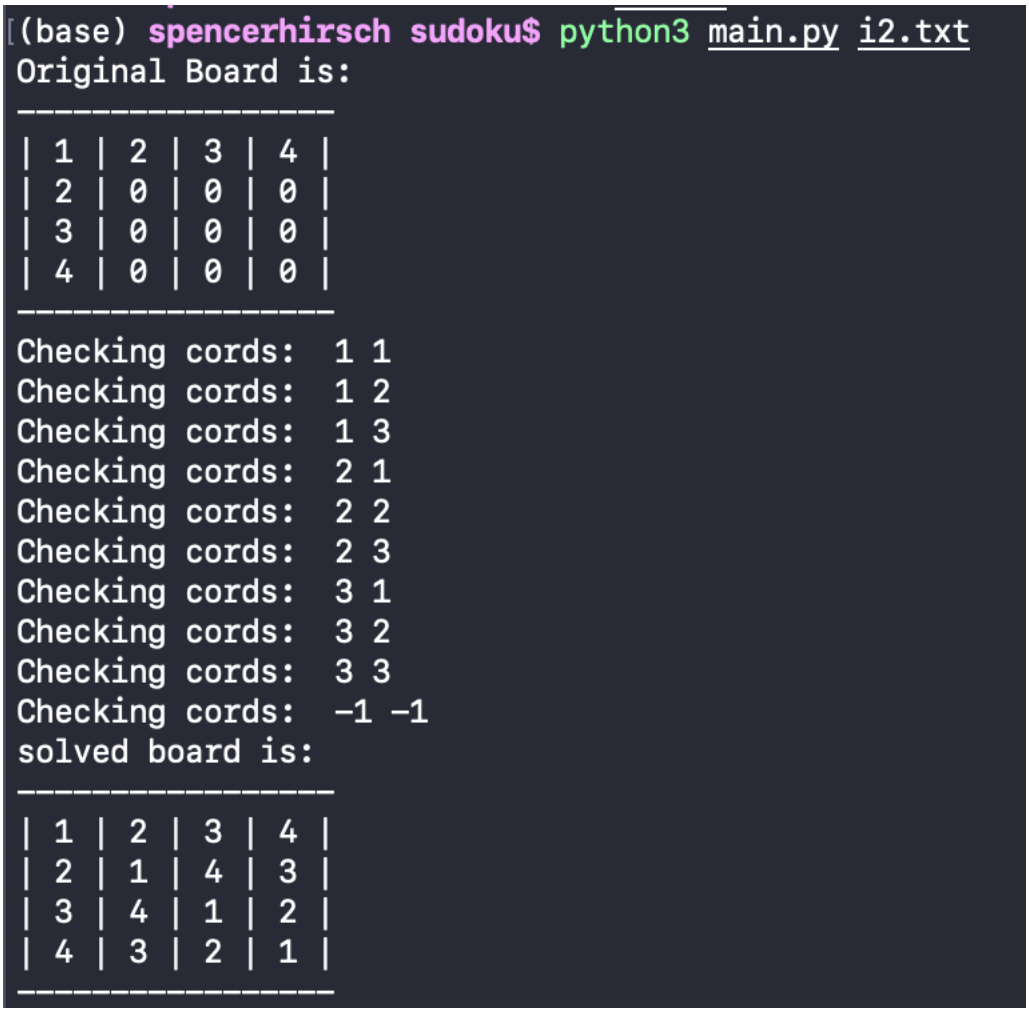
Screenshot demonstrating that the code runs.

Screenshot showing output for first test case:



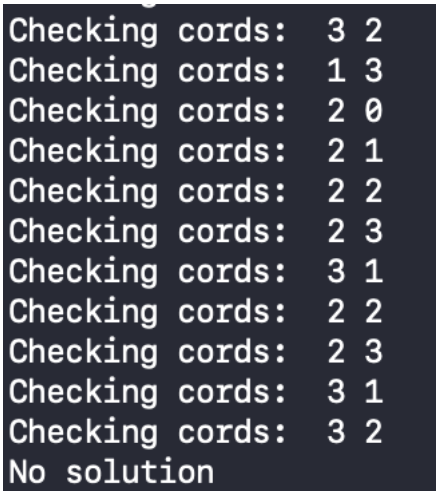
Screenshot demonstrating our first test case, as well as correct output.

Screenshot showing output for second test case:



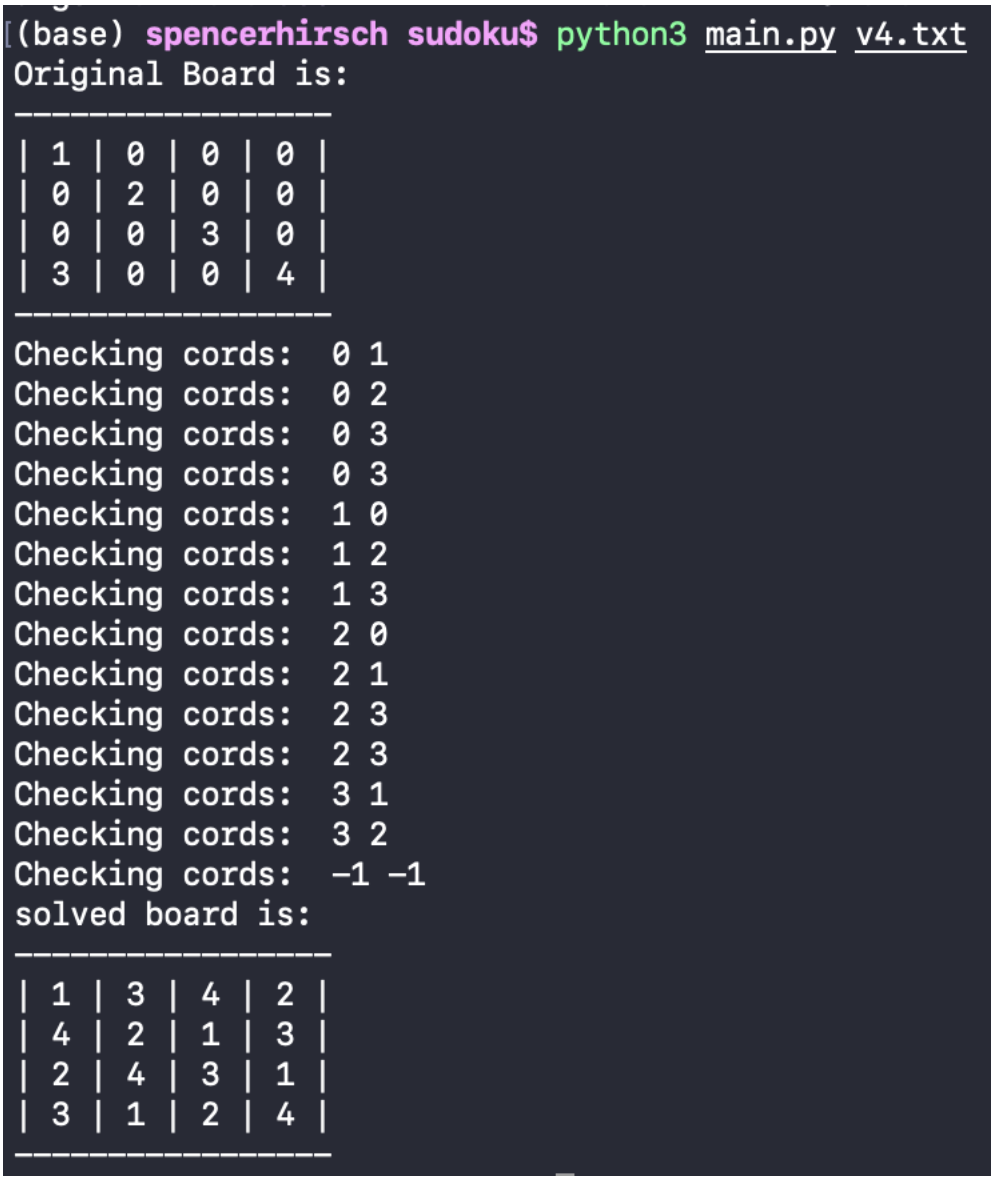
Screenshot demonstrating our second test case, as well as the correct output.

Screenshot showing output for third test case (impossible):



Screenshot demonstrating the output for an impossible puzzle given to the program.

Screenshot showing output for fourth test case:



Screenshot demonstrating the output for a fourth test case, for good measure.

Summary of the intermediate submission:

For this assignment, my partner and I used the backtracking algorithm in order to solve for the problem. Our program reads in a text file that conatins the number of rows and columns as well as a list of the values that the matrix will be made up of. The values will be read in by row,

$$R_{00}, R_{01}, R_{02}, R_{03}, R_{10}, R_{11}, R_{12}, R_{13}, R_{20}, R_{21}, R_{22}, R_{23}, R_{30}, R_{31}, R_{32}, R_{33}$$

The values are then placed in their repsective places in the n $\times$ n matrix. The initial empty spaces hold a value of 0, the algorithm will search for the 0's in the matrix and replace them with their correct value. We chose to use a backtracking algorithm in order to solve this problem. The psuedo-code for our solution is posted above. The solution that we came to is our original code.