Quad Layout TypeScript 14px Sublime

Prompt

Run Code

Solution 1

Difficulty: **Category:** Recursion **Successful Submissions:** 1,384+

Solve Sudoku ☆

You're given a two-dimensional array that represents a 9x9 partially filled Sudoku board. Write a function that returns the solved Sudoku board.

Sudoku is a famous number-placement puzzle in which you need to fill a 9x9 grid with integers in the range of `1-9`. Each 9x9 Sudoku board is split into 9 3x3 subgrids, as seen in the illustration below, and starts out partially filled.

```
- - 3 | - 2 - | 6 - -
9 - - | 3 - 5 | - - 1
- - 1 | 8 - 6 | 4 - -
- - - - - - - - - - -
- - 8 | 1 - 2 | 9 - -
7 - - | - - - | - - 8
- - 6 | 7 - 8 | 2 - -
- - - - - - - - - - -
- - 2 | 6 - 9 | 5 - -
8 - - | 2 - 3 | - - 9
- - 5 | - 1 - | 3 - -
```

The objective is to fill the grid such that each row, column, and 3x3 subgrid contains the numbers `1-9` exactly once. In other words, no row may contain the same digit more than once, no column may contain the same digit more than once, and none of the 9 3x3 subgrids may contain the same digit more than once.

Your input for this problem will always be a partially filled 9x9 two-dimensional array that represents a solvable Sudoku puzzle. Every element in the array will be an integer in the range of `0-9`, where a `0` represents an empty square that must be filled by your algorithm.

Note that you may modify the input array and that there will always be exactly one solution to each input Sudoku board.

Sample Input

```
board =
[
  [7, 8, 0, 4, 0, 0, 1, 2, 0],
```

```
    for
      if

      }
    }

    boar
    retu
  }

  const
    cons
    cons

    if (

    // c
    for
      fo
  [6, 0, 0, 0, 7, 5, 0, 0, 9],
  [0, 0, 0, 6, 0, 1, 0, 7, 8],
  [0, 0, 7, 0, 4, 0, 2      0],
  [0, 0, 1, 0, 5, 0, 9, 3, 0],
  [9, 0, 4, 0, 6, 0, 0, 0, 5],
  [0, 7, 0, 3, 0, 0, 0, 1,
  [1, 2, 0, 0, 0, 7, 4, 0,
  [0, 4, 9, 2, 0, 6, 0, 0, 7],
]
```

New AlgoExpert Question!

Read More

Close

```
      }
    }

    retu
  }
```

Sample Output

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

Submit Code

Co

Your sub

ou

Hints

Hint 1

Hint 2

Hint 3

Optimal Space & Time Complexity