

```

# sudoku_solver.py
#source venv/bin/activate

import random
from constraint import Problem, AllDifferentConstraint

def sudoku_solver(puzzle):
    """
    Solves a 9x9 Sudoku puzzle using python-constraint.
    Returns a random valid solution by shuffling domains.
    """
    problem = Problem()

    for row in range(9):
        for col in range(9):
            cell = (row, col)
            if puzzle[row][col] == 0:
                domain = list(range(1, 10))
                random.shuffle(domain) # Shuffle to vary solution path
                problem.addVariable(cell, domain)
            else:
                problem.addVariable(cell, [puzzle[row][col]])

    for row in range(9):
        problem.addConstraint(AllDifferentConstraint(), [(row, col) for col in range(9)])

    for col in range(9):
        problem.addConstraint(AllDifferentConstraint(), [(row, col) for row in range(9)])

    for box_row in range(3):
        for box_col in range(3):
            cells = [(box_row * 3 + r, box_col * 3 + c) for r in range(3) for c in range(3)]
            problem.addConstraint(AllDifferentConstraint(), cells)

    solution_iter = problem.getSolutionIter()
    for solution in solution_iter:
        solved_grid = [[solution[(row, col)] for col in range(9)] for row in range(9)]
        return solved_grid
    return None

def print_grid(grid):
    for row in grid:
        print(" ".join(str(num) for num in row))

if __name__ == "__main__":
    puzzle = [
        [0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 8, 0, 0, 0, 0, 0], # One clue to reduce search space
        [0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0]
    ]

```

```
]

solution = sudoku_solver(puzzle)
if solution:
    print("Solved Sudoku:")
    print_grid(solution)
else:
    print("No solution found.")
```