# 7/20/24, 6:36 PM Task3.ipynb - Colab

**Name:Sujit S Jayappa**

**TrackCode:SD**

**Task3:Implement a Sudoku Solver**

1. def print\_board(board):
2. for i in range(len(board)):
3. if i % 3 == 0 and i != 0:
4. print("- - - - - - - - - - - - - ")

5

6 for j in range(len(board[0])): 7 if j % 3 == 0 and j != 0:

8 print(" | ", end="") 9

1. if j == 8:
2. print(board[i][j]) 12 else:

13 print(str(board[i][j]) + " ", end="")

14

1. def find\_empty\_location(board):
2. for row in range(len(board)):
3. for col in range(len(board[0])):
4. if board[row][col] == 0:
5. return (row, col)
6. return None

21

1. def is\_valid(board, num, pos):
2. # Check row
3. for i in range(len(board[0])):
4. if board[pos[0]][i] == num and pos[1] != i:
5. return False

27

1. # Check column
2. for i in range(len(board)):
3. if board[i][pos[1]] == num and pos[0] != i:
4. return False

32

1. # Check 3x3 box
2. box\_x = pos[1] // 3
3. box\_y = pos[0] // 3

36

1. for i in range(box\_y \* 3, box\_y \* 3 + 3):
2. for j in range(box\_x \* 3, box\_x \* 3 + 3):
3. if board[i][j] == num and (i, j) != pos:
4. return False

41

42 return True

43

1. def solve\_sudoku(board):
2. find = find\_empty\_location(board) 46 if not find: 47 return True 48 else:

49 row, col = find

50

1. for i in range(1, 10):
2. if is\_valid(board, i, (row, col)):
3. board[row][col] = i

54

1. if solve\_sudoku(board):
2. return True

57

58 board[row][col] = 0

59

60 return False 61

1. # Example usage:
2. if \_\_name\_\_ == "\_\_main\_\_":
3. # Example board to solve
4. board = [
5. [5, 3, 0, 0, 7, 0, 0, 0, 0],
6. [6, 0, 0, 1, 9, 5, 0, 0, 0],
7. [0, 9, 8, 0, 0, 0, 0, 6, 0],
8. [8, 0, 0, 0, 6, 0, 0, 0, 3],
9. [4 0 0 8 0 3 0 0 1]

# https://colab.research.google.com/drive/1iwg3ataYhmj\_J2cS3-p8Mg6tiBGR6Cqs#scrollTo=4IU1S0--NCkb 1/2 7/20/24, 6:36 PM Task3.ipynb - Colab

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

76

1. print("Sudoku board to solve:")
2. print\_board(board)

79

1. if solve\_sudoku(board):
2. print("\nSolution:") 82 print\_board(board) 83 else:

84 print("\nNo solution exists.")

85

Sudoku board to solve:

1. 3 0 | 0 7 0 | 0 0 0
2. 0 0 | 1 9 5 | 0 0 0

0 9 8 | 0 0 0 | 0 6 0

- - - - - - - - - - - - -

8 0 0 | 0 6 0 | 0 0 3

4 0 0 | 8 0 3 | 0 0 1

1. 0 0 | 0 2 0 | 0 0 6

- - - - - - - - - - - - -

0 6 0 | 0 0 0 | 2 8 0

0 0 0 | 4 1 9 | 0 0 5

0 0 0 | 0 8 0 | 0 7 9

Solution:

1. 3 4 | 6 7 8 | 9 1 2
2. 7 2 | 1 9 5 | 3 4 8

1 9 8 | 3 4 2 | 5 6 7

- - - - - - - - - - - - -

1. 5 9 | 7 6 1 | 4 2 3

4 2 6 | 8 5 3 | 7 9 1

7 1 3 | 9 2 4 | 8 5 6

- - - - - - - - - - - - -

1. 6 1 | 5 3 7 | 2 8 4
2. 8 7 | 4 1 9 | 6 3 5
3. 4 5 | 2 8 6 | 1 7 9

# https://colab.research.google.com/drive/1iwg3ataYhmj\_J2cS3-p8Mg6tiBGR6Cqs#scrollTo=4IU1S0--NCkb 2/2