Name:Vrushali Magdum.

TrackCode: SD

Task3: Implement a Sudoku Solver

```
def print_board(board):
    for i in range(len(board)):
        if i % 3 == 0 and i != 0:
            print(" ---------------------------------------")

        for j in range(len(board[0])):
            if j % 3 == 0 and j != 0:
                print(" | ", end="")

            if j == 8:
                print(board[i][j])
            else:
                print(str(board[i][j]) + " ", end="")

def find_empty_location(board):
    for row in range(len(board)):
        for col in range(len(board[0])):
            if board[row][col] == 0:
                return (row, col)
    return None

def is_valid(board, num, pos):
    # Check row
    for i in range(len(board[0])):
        if board[pos[0]][i] == num and pos[1] != i:
            return False

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

    # Check 3x3 box
    box_x = pos[1] // 3
    box_y = pos[0] // 3

    for i in range(box_y * 3, box_y * 3 + 3):
        for j in range(box_x * 3, box_x * 3 + 3):
            if board[i][j] == num and (i, j) != pos:
                return False

    return True

def solve_sudoku(board):
    find = find_empty_location(board)
    if not find:
        return True
    else:
        row, col = find

    for i in range(1, 10):
        if is_valid(board, i, (row, col)):
            board[row][col] = i

            if solve_sudoku(board):
                return True

            board[row][col] = 0

    return False

# Example usage:
if __name__ == "__main__":

    # Example board to solve
    board = [
        [5, 3, 0, 0, 7, 0, 0, 0, 0],
        [6, 0, 0, 1, 9, 5, 0, 0, 0],
        [0, 9, 8, 0, 0, 0, 0, 6, 0],
```

```
69          [8, 0, 0, 0, 6, 0, 0, 0, 3],
```

```
        [4, 0, 0, 8, 0, 3, 0, 0, 1],
        [7, 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]
    ]


    print("Sudoku board to solve:")
    print_board(board)

    if solve_sudoku(board):
        print("\nSolution:")
        print_board(board)
    else:
        print("\nNo solution exists.")

```

```
Sudoku board to solve:
5 3 0  | 0 7 0  | 0 0 0
6 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
7 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:
5 3 4  | 6 7 8  | 9 1 2
6 7 2  | 1 9 5  | 3 4 8
1 9 8  | 3 4 2  | 5 6 7
- - - - - - - - - - - -
8 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
- - - - - - - - - - - -
9 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
```