**Create a program that solves Sudoku puzzles automatically.**

def print\_grid(grid):

for row in grid:

print(" ".join(str(num) if num != 0 else "." for num in row))

def is\_valid\_move(grid, row, col, num):

# Check if num is not in the current row, column, or 3x3 sub-grid

for i in range(9):

if grid[row][i] == num or grid[i][col] == num:

return False

start\_row, start\_col = 3 \* (row // 3), 3 \* (col // 3)

for i in range(3):

for j in range(3):

if grid[start\_row + i][start\_col + j] == num:

return False

return True

def find\_empty\_location(grid):

# Find an empty cell (represented by 0)

for row in range(9):

for col in range(9):

if grid[row][col] == 0:

return row, col

return None

def solve\_sudoku(grid):

empty\_loc = find\_empty\_location(grid)

if not empty\_loc:

return True # Puzzle solved

row, col = empty\_loc

for num in range(1, 10):

if is\_valid\_move(grid, row, col, num):

grid[row][col] = num

if solve\_sudoku(grid):

return True

grid[row][col] = 0 # Backtrack

return False

def main():

# Example unsolved Sudoku puzzle

sudoku\_grid = [

[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]

]

print("Unsolved Sudoku:")

print\_grid(sudoku\_grid)

if solve\_sudoku(sudoku\_grid):

print("\nSolved Sudoku:")

print\_grid(sudoku\_grid)

else:

print("No solution exists.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

