Task 3

3.Create a program that solves Sudoku Puzzles automatically. The program should take an input grid representing an unsolved sudoku puzzle and use an algorithm to fill in the missing numbers

```
public class SudokuSolver {

    // Size of the grid
    static final int SIZE = 9;

    // Function to check if placing a number is valid
    static boolean isValid(int[][] board, int row, int col, int
      num) {
        // Check row and column
        for (int i = 0; i < SIZE; i++) {
            if (board[row][i] == num || board[i][col] == num)
                return false;
        }

        // Check 3x3 box
        int startRow = row - row % 3;
        int startCol = col - col % 3;
```

```
    for (int i = startRow; i < startRow + 3; i++) {
        for (int j = startCol; j < startCol + 3; j++) {
            if (board[i][j] == num)
                return false;
        }
    }

    return true;
}

// Backtracking function to solve Sudoku
static boolean solveSudoku(int[][] board) {
    for (int row = 0; row < SIZE; row++) {
        for (int col = 0; col < SIZE; col++) {
            if (board[row][col] == 0) {
                for (int num = 1; num <= SIZE; num++) {
                    if (isValid(board, row, col, num)) {
                        board[row][col] = num;

                        if (solveSudoku(board))
                            return true;
```

```
                    // Backtrack
                    board[row][col] = 0;
                }
            }
            return false; // trigger backtracking
        }
    }
    return true; // Solved
}

// Utility function to print the board
static void printBoard(int[][] board) {
    for (int row = 0; row < SIZE; row++) {
        for (int col = 0; col < SIZE; col++) {
            System.out.print(board[row][col] + " ");
        }
        System.out.println();
    }
}
```

```
    // Main function with sample puzzle
    public static void main(String[] args) {
        int[][] 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},
            {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}
        };

        if (solveSudoku(board)) {
            System.out.println("Solved Sudoku:");
            printBoard(board);
        } else {
            System.out.println("No solution exists.");
        }
    }
}
```