**Valid Sudoku**

bool isValid(int grid[9][9], int num, int row, int col) {

// Checking row and column

for (int k = 0; k < 9; k++) {

if (grid[row][k] == num || grid[k][col] == num) {

return false;

}

}

// Checking 3x3 subgrid

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 (grid[i][j] == num) {

return false;

}

}

}

return true;

}

bool solveSudoku(int grid[9][9], int row, int col) {

if (row == 9) {

return true;

}

if (col == 9) {

return solveSudoku(grid, row + 1, 0);

}

if (grid[row][col] != 0) {

return solveSudoku(grid, row, col + 1);

}

for (int num = 1; num <= 9; num++) {

if (isValid(grid, num, row, col)) {

grid[row][col] = num;

if (solveSudoku(grid, row, col + 1)) {

return true;

}

// Otherwise, backtrack

grid[row][col] = 0;

}

}

return false;

}

bool isItSudoku(int matrix[9][9]) {

if (solveSudoku(matrix, 0, 0)) {

return true;

}

return false;

}