```
#include <bits/stdc++.h>
using namespace std;

int board[4][4];

bool isSafe(int row, int col, int val) {
    // Checking row and column
    for (int i = 0; i < 4; i++) {
        if (board[row][i] == val || board[i][col] == val) return false;
    }

    // Checking the 2x2 subgrid
    int startRow = row - row % 2;
    int startCol = col - col % 2;
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            if (board[startRow + i][startCol + j] == val) return false;
        }
    }

    return true;
}

bool solve() {
    int n = 4;
    for (int row = 0; row < n; row++) {
        for (int col = 0; col < n; col++) {
            if (board[row][col] == 0) { // Find an empty cell
                for (int val = 1; val <= 4; val++) { // Trying numbers from 1 to 9
                    if (isSafe(row, col, val)) {
```

```
                board[row][col] = val; // insert the number
                if (solve()) return true; // Recursively solve for the rest
                board[row][col] = 0; // Backtrack if solution not possible
            }
        }
        return false; // If no valid number can be placed, return false
      }
    }
  }
  return true; // Solved
}

int main() {
  cout << "Enter Sudoku: 2*2 (use 0 for empty cells and 2*2 sudoku means: 4*4 2D matrix):
\n";
  for (int i = 0; i < 4; i++) {
    for (int j = 0; j < 4; j++) {
      cin >> board[i][j];
    }
  }

  if (solve()) {
    cout << "Solved Sudoku:\n";
    for (int i = 0; i < 4; i++) {
      for (int j = 0; j < 4; j++) {
        cout << board[i][j] << " ";
      }
      cout << endl;
    }
  } else {
    cout << "\nSolution not possible!\n";
```

```
    }

    return 0;
}
```