Experiment 9

Student Name: Harsh Kumar **UID: 22BCS15754**
Branch: BE-CSE **Section/Group: 603_FL_IOT-B**
Semester: 5th **Date of Performance:28/10/2024**
Subject Name: AP **Subject Code: 22CSP-314**

1. Aim: Given a partially filled 9 x 9 sudoku board, you need to output a solution to it.
Input:
The first line contains N which is the number of Sudoku puzzles. N Sudoku puzzles follow. Each one contains 9 lines with each line containing 9 space separated integers. A 0 indicates an unfilled square, and an integer between 1 and 9 denotes a filled square with that value.

2. Objective: The objective of this program is to **solve multiple Sudoku puzzles** by filling in the missing numbers in a given 9x9 grid such that each row, each column, and each of the nine 3x3 sub-grids contains all the numbers from 1 to 9 exactly once.

3. Implementation/Code:

```
#include <iostream>
#include <vector>
using namespace std;
bool isValid(vector<vector<int>>& board, int row, int col, int num) {
    int subRow = 3 * (row / 3);
    int subCol = 3 * (col / 3);
    for (int i = 0; i < 9; i++) {
        if (board[row][i] == num || board[i][col] == num || board[subRow + i /
3][subCol + i % 3] == num) {
            return false;}}
    return true;}
bool solveSudoku(vector<vector<int>>& board) {
    for (int row = 0; row < 9; row++) {
        for (int col = 0; col < 9; col++) {
            if (board[row][col] == 0) {
                for (int num = 1; num <= 9; num++) {
                    if (isValid(board, row, col, num)) {
                        board[row][col] = num;
                        if (solveSudoku(board))
                            return true;
```

```
                    board[row][col] = 0 } }
                return false;}} }
        return true}
    void printBoard(const vector<vector<int>>& board) {
        for (const auto& row : board) {
            for (int num : row) {
                cout << num << " ";}
            cout << endl;}}
    int main() {
        vector<vector<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} };
        solveSudoku(board);
        printBoard(board);
        return 0;
        }
```

4. OUTPUT:

```
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


...Program finished with exit code 0
Press ENTER to exit console.
```