1

```c
#include <stdio.h>
#include <stdbool.h>

#define SIZE 9

bool isSafe(int row, int col, int board[SIZE][SIZE], int val) {
    for (int i = 0; i < SIZE; i++) {
        // Row checking
        if (board[row][i] == val)
            return false;
        // Column checking
        if (board[i][col] == val)
            return false;
        // 3x3 matrix checking
        if (board[3 * (row / 3) + i / 3][3 * (col / 3) + i % 3] == val)
            return false;
    }
    return true;
}
```

2.

```
bool solve(int board[SIZE][SIZE]) {
    for (int row = 0; row < SIZE; row++) {
        for (int col = 0; col < SIZE; col++) {
            // Cell empty
            if (board[row][col] == 0) {
                for (int val = 1; val <= 9; val++) {
                    if (isSafe(row, col, board, val)) {
                        board[row][col] = val;
                        // Recursive call
                        bool ifPossible = solve(board);
                        if (ifPossible)
                            return true;
                        else {
                            // Backtracking
                            board[row][col] = 0;
                        }
                    }
                }
                return false;
            }
        }
    }
    return true;
}
```

3

```c
void print(int board[SIZE][SIZE]) {
    printf("\n\n");
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            printf("%d ", board[i][j]);
        }
        printf("\n");
    }
    printf("\n\n");
}
```

```c
int main() {
    printf("WELCOME TO SUDOKU SOLVER\n\n");
    printf("Instructions:\n\n");
    printf("Sudoku is a popular number puzzle game that involves filling a 9x9 grid with numbers.\n");
    printf("The grid is divided into nine 3x3 subgrids, and the objective is to fill each row, column,
    printf("However, no number can be repeated in any row, column, or subgrid.\n\n");

    printf("Let's start the play\n\n");

    int sudoku[SIZE][SIZE];

    printf("Enter the difficulty level:\n");
    printf("Ranging from 1 to 3\n");

    int x;
    scanf("%d", &x);

    // Insert the question vector based on difficulty level
    if (x == 1) {
        int ques[] = {3, 1, 6, 5, 0, 8, 4, 0, 2, 5, 2, 0, 1, 3, 4, 0, 6, 8, 4, 8, 7, 6, 2, 9, 5, 3, 1,

        // Store the question values in the Sudoku board
        int cnt = 0;
        for (int i = 0; i < SIZE; i++) {
            for (int j = 0; j < SIZE; j++) {
                sudoku[i][j] = ques[cnt++];
            }
        }
    } else if (x == 2) {
        int ques[] = {3, 1, 6, 5, 7, 8, 4, 0, 2, 5, 2, 0, 1, 3, 4, 0, 6, 8, 4, 8, 7, 6, 2, 9, 5, 3, 1,
```

```
    // Store the question values in the Sudoku board
    int cnt = 0;
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            sudoku[i][j] = ques[cnt++];
        }
    }
} else if (x == 3) {
    int ques[] = {3, 0, 6, 5, 0, 8, 4, 0, 0, 5, 2, 0, 0, 0, 0, 0, 0, 0, 0, 8, 7, 0, 0, 0, 0, 3, 1, 0,

    // Store the question values in the Sudoku board
    int cnt = 0;
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            sudoku[i][j] = ques[cnt++];
        }
    }
} else {
    printf("Invalid difficulty option!!!!\n");
    return 0;
}

// Given Sudoku
print(sudoku);

// User interface part
printf("Provide your solution Sudoku (in a 2D matrix):\n");
int userSolution[SIZE][SIZE];
```

```c
// User interface part
printf("Provide your solution Sudoku (in a 2D matrix):\n");
int userSolution[SIZE][SIZE];

// Taking inputs from the user
for (int i = 0; i < SIZE; i++) {
    for (int j = 0; j < SIZE; j++) {
        scanf("%d", &userSolution[i][j]);
    }
}

solve(sudoku);

// User interface part (again)
int isEqual = 1; // Flag to check if the user's solution is correct

// Compare the user's solution with the solved Sudoku
for (int i = 0; i < SIZE; i++) {
    for (int j = 0; j < SIZE; j++) {
        if (sudoku[i][j] != userSolution[i][j]) {
            isEqual = 0;
            break;
        }
    }
    if (!isEqual) {
        break;
    }
}

if (isEqual) {
    printf("Your provided Sudoku solution is CORRECT!\n\n");
```

```
    if (isEqual) {
        printf("Your provided Sudoku solution is CORRECT!\n\n");
    } else {
        printf("Your provided Sudoku solution isn't CORRECT!\n\n");
        printf("Here is the solution of the Sudoku:\n\n");
        // Printing the solved Sudoku for the wrong answer
        print(sudoku);
    }

    return 0;
}
```

7