SANYAM AGRAWAL – SE21UCSE192 – CSE3

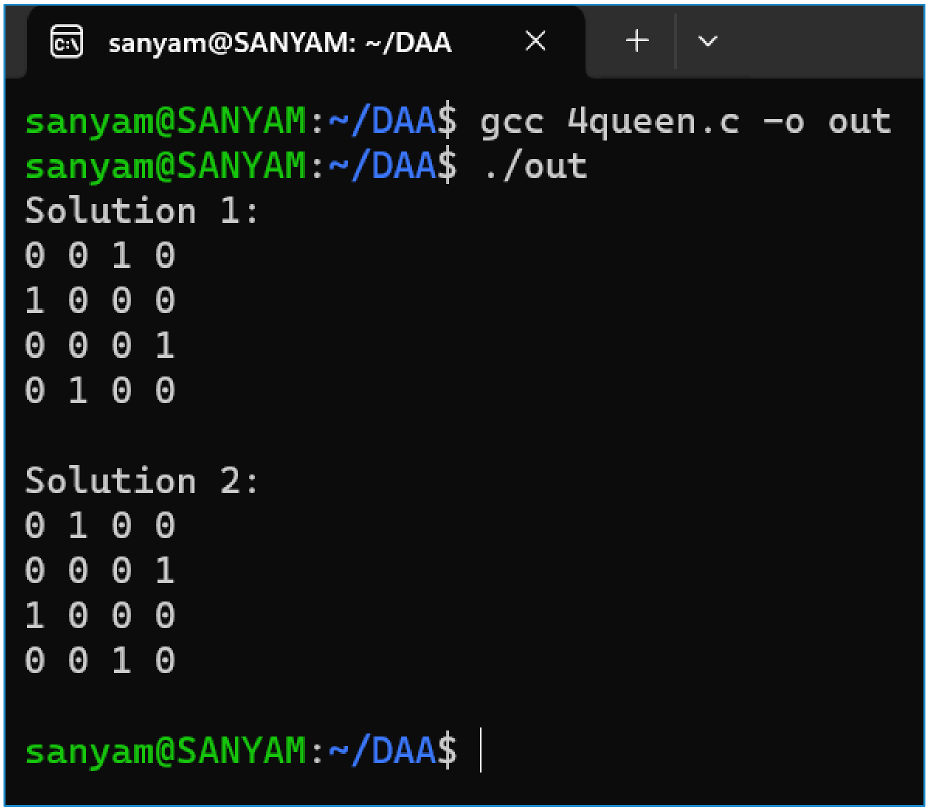
DAA Lab Assignment 12 (4 – Queen)

Source Code : ->

```
#include <stdio.h>

#include <stdbool.h>


#define N 4


void printSolution(int board[N][N], int solutionNumber) {
  printf("Solution %d:\n", solutionNumber);
  for (int i = 0; i < N; i++) {
    for (int j = 0; j < N; j++) {
      printf("%d ", board[i][j]);
    }
    printf("\n");
  }
  printf("\n");
}


bool isSafe(int board[N][N], int row, int col) {
  for (int i = 0; i < col; i++) {
    if (board[row][i]) {
      return false;
    }
  }
  for (int i = row, j = col; i >= 0 && j >= 0; i--, j--) {
    if (board[i][j]) {
            return false;
        }
    }
    for (int i = row, j = col; i < N && j >= 0; i++, j--) {
        if (board[i][j]) {
            return false;
        }
    }

    return true;
}

bool solveNQueensUtil(int board[N][N], int col, int *solutionCount) {
    if (col >= N) {
        (*solutionCount)++;
        printSolution(board, *solutionCount);
        return true;
    }

    bool res = false;

    for (int i = 0; i < N; i++) {
        if (isSafe(board, i, col)) {
            board[i][col] = 1;
            res = solveNQueensUtil(board, col + 1, solutionCount) || res;
            board[i][col] = 0;
        }
    }
    return res;
```

```
}

void solveNQueens() {
    int board[N][N] = {{0}};
    int solutionCount = 0;

    if (!solveNQueensUtil(board, 0, &solutionCount)) {
        printf("Solution does not exist.\n");
    }
}
int main() {
    solveNQueens();
    return 0;
}
```

Output Screenshot:->



```
sanyam@SANYAM:~/DAA$ gcc 4queen.c -o out
sanyam@SANYAM:~/DAA$ ./out
Solution 1:
0 0 1 0
1 0 0 0
0 0 0 1
0 1 0 0

Solution 2:
0 1 0 0
0 0 0 1
1 0 0 0
0 0 1 0

sanyam@SANYAM:~/DAA$
```