Exp_10\N_Queen.c

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

bool isSafe(int** board, int row, int col, int n) {
    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 solveNQueenUtil(int** board, int col, int n) {
    if (col >= n)
        return true;

    for (int i = 0; i < n; i++) {
        if (isSafe(board, i, col, n)) {
            board[i][col] = 1;

            if (solveNQueenUtil(board, col + 1, n))
                return true;

            board[i][col] = 0;
        }
    }

    return false;
}

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

bool solveNQueen(int n) {
    int** board = (int**)malloc(n * sizeof(int*));
    for (int i = 0; i < n; i++) {
        board[i] = (int*)calloc(n, sizeof(int));
    }
```

```
52
53      if (!solveNQueenUtil(board, 0, n)) {
54          printf("Solution does not exist\n");
55          for (int i = 0; i < n; i++)
56              free(board[i]);
57          free(board);
58          return false;
59      }
60
61      printSolution(board, n);
62
63      for (int i = 0; i < n; i++)
64          free(board[i]);
65      free(board);
66      return true;
67  }
68
69  int main() {
70      int n;
71      printf("Enter the value of N: ");
72      scanf("%d", &n);
73
74      solveNQueen(n);
75
76      return 0;
77  }
```