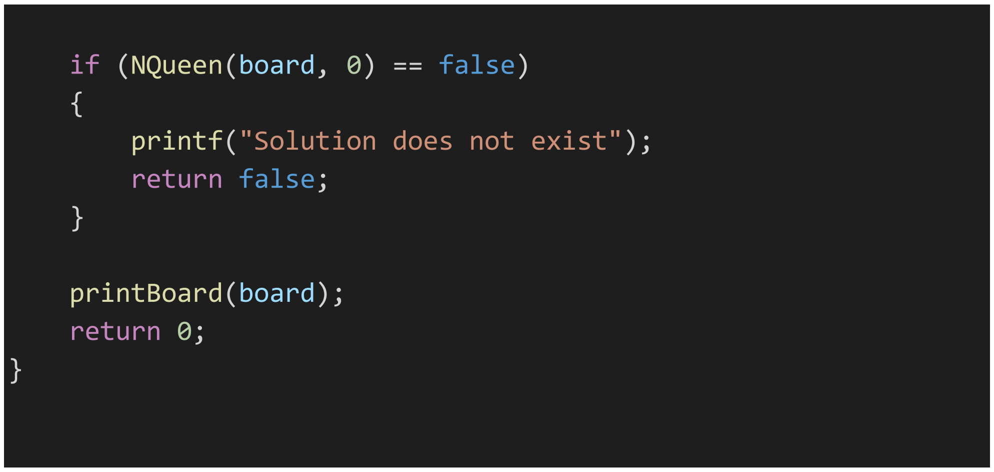
Name	Siddhesh Sonar
UID no.	2021700063
Experiment No.	7

AIM:	To implement N Queens problem using backtracking
Program	
PROBLEM STATEMENT :	To implement N Queens problem using backtracking
ALGORITHM/ THEORY:	Initialize an empty chessboard of size NxN. Start with the leftmost column and place a queen in the first row of that column. Move to the next column and place a queen in the first row of that column. Repeat step 3 until either all N queens have been placed or it is impossible to place a queen in the current column without violating the rules of the problem. If all N queens have been placed, print the solution. If it is not possible to place a queen in the current column without violating the rules of the problem, backtrack to the previous column. Remove the queen from the previous column and move it down one row. Repeat steps 4-7 until all possible configurations have been tried.

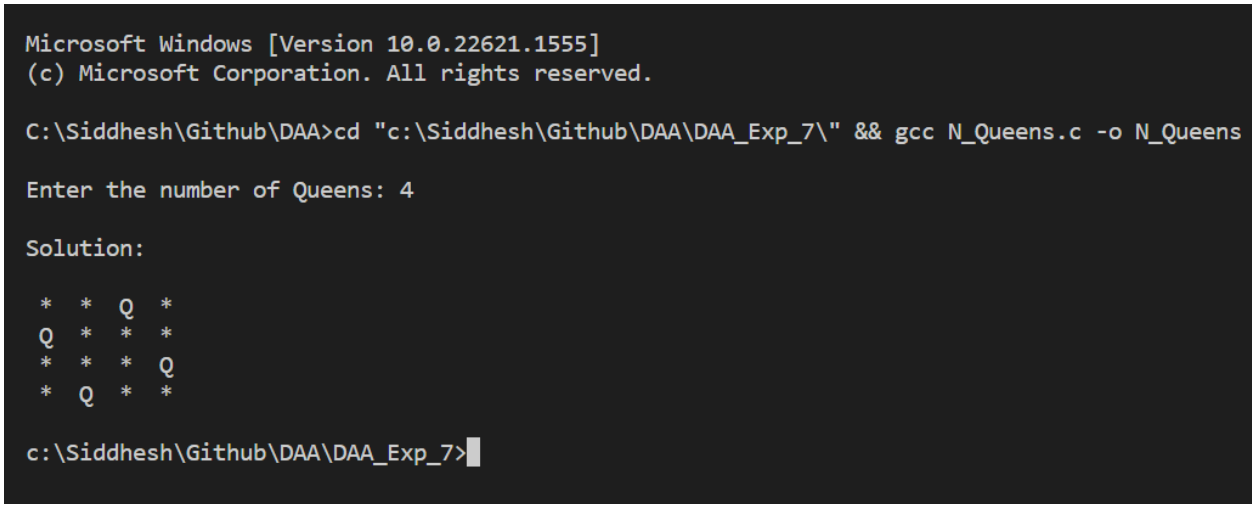
PROGRAM:

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

int n;

void printBoard(int board[n][n])
{
    printf("\nSolution:\n\n");
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (board[i][j] == 1)
            {
                printf(" Q ");
            }
            else
            {
                printf(" * ");
            }
        }
        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; j >= 0 && i < n; i++, j--)
    {
        if (board[i][j])
        {
            return false;
        }
    }

    return true;
}

bool NQueen(int board[n][n], int col)
{
    if (col >= n)
    {
        return true;
    }
    for (int i = 0; i < n; i++)
    {
        if (isSafe(board, i, col))
        {
            board[i][col] = 1;
            if (NQueen(board, col + 1))
            {
                return true;
            }

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

int main()
{
    printf("\nEnter the number of Queens: ");
    scanf("%d", &n);
    int board[n][n];
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            board[i][j] = 0;
        }
    }
```



```c
    if (NQueen(board, 0) == false)
    {
        printf("Solution does not exist");
        return false;
    }

    printBoard(board);
    return 0;
}
```

RESULT:



```
Microsoft Windows [Version 10.0.22621.1555]
(c) Microsoft Corporation. All rights reserved.

C:\Siddhesh\Github\DAA>cd "c:\Siddhesh\Github\DAA\DAA_Exp_7\" && gcc N_Queens.c -o N_Queens

Enter the number of Queens: 4

Solution:

 *  *  Q  *
 Q  *  *  *
 *  *  *  Q
 *  Q  *  *

c:\Siddhesh\Github\DAA\DAA_Exp_7>
```

CONCLUSION: Successfully understood NQueens algorithm and implemented it in C program.