Roll. No. A016	Name: Varun Khadayate
Class B.Tech CsBs	Batch: 1
Date of Experiment: 7-2-2021	Date of Submission: 21-2-2022

To implement N Queens Problem

The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. For example, following is a solution for 4 Queen problem.
The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. For example, following are two solutions for 4 Queen problem.

Algorithm

The idea is to place queens one by one in different columns, starting from the leftmost column. When we place a queen in a column, we check for clashes with already placed queens. In the current column, if we find a row for which there is no clash, we mark this row and column as part of the solution. If we do not find such a row due to clashes then we backtrack and return false.

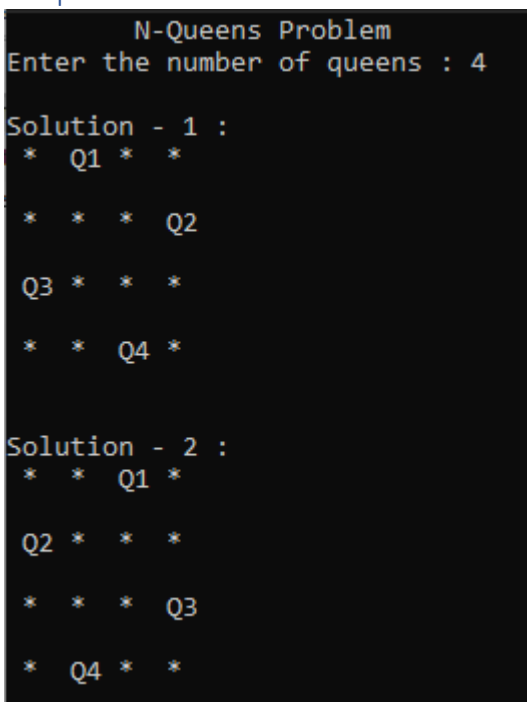
1) Start in the leftmost column
2) If all queens are placed return true
3) Try all rows in the current column. Do following for every tried row.
   a) If the queen can be placed safely in this row then mark this [row, column] as part of the solution and recursively check if placing queen here leads to a solution.
   b) If placing queen in [row, column] leads to a solution then return true.
   c) If placing queen doesn't lead to a solution then unmark this [row, column] (Backtrack) and go to step (a) to try other rows.
4) If all rows have been tried and nothing worked, return false to trigger backtracking.

Code

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

int NoSoln(int k, int col[])
{
   int i;
   for(i=1;i<=k-1;i++)
   {
      if(col[k]==col[i] || (abs(i-k)==abs(col[i] - col[k])))
         return 1;
   }
   return 0;
}


int NQueen(int n)
{
   int k = 1;
   int count=0;
   int i,j,col[n+1];
   col[k]=0;
   while(k!=0)
```

```
    {
        col[k] += 1;
        while(col[k]<=n && NoSoln(k,col))
            col[k]=col[k]+1;
        if(col[k]<=n)
        {
            if(k==n)
            {
                count++;
                printf("\nSolution - %d : \n",count);
                for(i=1;i<=n;i++)
                {
                    for(j=1;j<=n;j++)
                        if(col[i] == j)
                            printf(" Q%d",i);
                        else
                            printf(" * ");
                    printf("\n\n");
                }
            }
        else
        {
            k++;
            col[k]=0;
        }
    }
    else
        k--;
    }
    return count;
}

int main()
{
    int n,solutions;
    printf("\tN-Queens Problem");
    printf("\nEnter the number of queens : ");
    scanf("%d",&n);
    solutions=NQueen(n);
    if(solutions==0)
        printf("No solution!!");
    return 0;
}
```

Output


N-Queens Problem
Enter the number of queens : 4
Solution - 1 :
* Q1 * *
* * * Q2
Q3 * * *
* * Q4 *
Solution - 2 :
* * Q1 *
Q2 * * *
* * * Q3
* Q4 * *