Sardar Patel Institute of Technology
Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058, India
(Autonomous College Affiliated to University of Mumbai)

Experiment No.	7
Aim	Implementing N Queens Problem
Name	Aarush Kinhikar
UID No.	2021300063
Class & Division	SE Comps A, Batch D

Theory:

The N-Queens problem is a classic combinatorial problem that involves placing N chess queens on an N×N chessboard in such a way that no two queens threaten each other, i.e., no two queens share the same row, column, or diagonal. The goal is to find all distinct arrangements of queens on the chessboard that satisfy this constraint.

The problem is called the "N-Queens" problem because it is typically stated in terms of finding solutions for different board sizes, where N represents the number of queens to be placed on the board. The most common variant is the 8-Queens problem, which involves placing 8 queens on an 8x8 chessboard.

The N-Queens problem is known to be an NP-complete problem, which means that it is difficult to find an efficient algorithm to solve it for large values of N. However, there are various techniques, such as backtracking, recursive algorithms, and optimization methods, that can be used to find solutions for smaller values of N or approximate solutions for larger values of N. The N-Queens problem has applications in computer science, artificial intelligence, and operations research, and it is often used as a benchmark problem to test the performance of algorithms in these fields.

Algorithm:

1. Start with an empty N×N chessboard.
2. Place the first queen in the first column of the first row.
3. Move on to the next row and try placing a queen in each column of that row, checking if it is safe to do so.
4. If a queen can be safely placed in a column of the current row, mark that cell as occupied by a queen and move on to the next row.
5. If no safe position is found in the current row, backtrack by undoing the last move and trying the next column in the previous row.
6. Continue this process until all N queens are placed on the board, or all possible placements are tried.
7. When a solution is found (i.e., all N queens are placed on the board without threatening each other), add it to the list of solutions.
8. Continue searching for more solutions, or backtrack further if necessary.
9. Once all solutions are found, the algorithm terminates.

Program:

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
int board[20],count;

int main(){
    int n,i,j;
    void queen(int row,int n);
    printf("N Queens Problem Using Backtracking");
    printf("\n\nEnter number of Queens:");
    scanf("%d",&n);
    queen(1,n);
    return 0;
}

void print(int n){
    int i,j;
    printf("\n\nSolution %d:\n\n",++count);
    for(i=1;i<=n;++i){
        printf("\t%d",i);
    }
    for(i=1;i<=n;++i){
```

```
        printf("\n\n%d",i);
        for(j=1;j<=n;++j){
            if(board[i]==j){
                printf("\tQ");
            }
            else{
                printf("\t-");
            }
        }
    }
}

int place(int row,int column){
    int i;
    for(i=1;i<=row-1;++i){
        if(board[i]==column){
            return 0;
        }
        else{
            if(abs(board[i]-column)==abs(i-row)){
                return 0;
            }
        }
    }
    return 1;
}

void queen(int row,int n){
    int column;
    for(column=1;column<=n;++column){
        if(place(row,column)){
            board[row]=column;
            if(row==n){
                print(n);
            }
            else{
                queen(row+1,n);
            }
        }
    }
}
```

Results:

For n=4,

```
Enter number of Queens:4

Solution 1:
        1       2       3       4
1       -       Q       -       -
2       -       -       -       Q
3       Q       -       -       -
4       -       -       Q       -
Solution 2:
        1       2       3       4
1       -       -       Q       -
2       Q       -       -       -
3       -       -       -       Q
4       -       Q       -       -
```

For n=8 (total 92 solutions),

```
Solution 91:
      1     2     3     4     5     6     7     8
1     -     -     -     -     -     -     -     Q
2     -     -     Q     -     -     -     -     -
3     Q     -     -     -     -     -     -     -
4     -     -     -     -     -     Q     -     -
5     -     Q     -     -     -     -     -     -
6     -     -     -     -     Q     -     -     -
7     -     -     -     -     -     -     Q     -
8     -     -     -     Q     -     -     -     -
Solution 92:
      1     2     3     4     5     6     7     8
1     -     -     -     -     -     -     -     Q
2     -     -     -     Q     -     -     -     -
3     Q     -     -     -     -     -     -     -
4     -     -     Q     -     -     -     -     -
5     -     -     -     -     -     Q     -     -
6     -     Q     -     -     -     -     -     -
```

Conclusion:

In conclusion, the backtracking algorithm is a popular and straightforward approach for solving the N-Queens problem, a classic and well-known combinatorial problem. By systematically placing queens on the chessboard and backtracking, when necessary, the algorithm can find all distinct arrangements of N queens on an N×N chessboard that satisfy the constraint of non-attacking queens. However, due to the NP-completeness of the problem, the backtracking algorithm may have exponential time complexity and may require further optimizations for large values of N.