AI Assignment 2 - Implementation of Uninformed Strategies

Name : Siddhi Rajeshirke
Division : CS-C
Batch : 3
Roll No : 75
PRN : 12110298

N Queens problem using DFS and Backtracking

Code :

```java
import java.util.Scanner;
public class nQueens {
    static int n;
    static void printSolution(int[][] board) {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                if (board[i][j] == 1) {
                    System.out.print(" Q ");
                } else {
                    System.out.print(" - ");
                }
            }
            System.out.println();
        }
    }
    static boolean isSafe(int[][] board, int row, int col) {
        for (int i = 0; i < col; i++) {
            if (board[row][i] == 1) {
                return false;
            }
        }
        for (int i = row, j = col; i >= 0 && j >= 0; i--, j--) {
            if (board[i][j] == 1) {
                return false;
            }
        }
        for (int i = row, j = col; i < n && j >= 0; i++, j--) {
            if (board[i][j] == 1) {
                return false;
            }
        }
        return true;
    }
    static boolean solveNQueensUtil(int[][] board, int col) {
        if (col >= n) {
            return true;
        }
        for (int i = 0; i < n; i++) {
            if (isSafe(board, i, col)) {
                board[i][col] = 1;

                if (solveNQueensUtil(board, col + 1)) {
                    return true;
                }
```

```
                board[i][col] = 0;
            }
        }
        return false;
    }
    static void solveNQueens(int a) {
        n = a;
        int[][] board = new int[n][n];

        if (!solveNQueensUtil(board, 0)) {
            System.out.println("Solution does not exist, for nQueens
problem enter n greater than 3");
        } else {
            printSolution(board);
        }
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the value of N : ");
        int a = scanner.nextInt();
        scanner.close();
        solveNQueens(a);
    }
}
```

Output :