```java
import java.util.Scanner;

public class Main {
    static int N;
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the size of the board (N): ");
        N = sc.nextInt();
        System.out.print("Enter the initial row and column for the first Queen:
        ");
        int firstQueenRow = sc.nextInt();
        int firstQueenCol = sc.nextInt();

        int[][] board = new int[N][N];
        board[firstQueenRow][firstQueenCol] = 1;

        if (!solveNQueensUtil(board, firstQueenRow + 1)) {
            System.out.println("No solution exists.");
        } else {
            printBoard(board);
        }
    }

    static boolean solveNQueensUtil(int[][] board, int row) {
        if (row == N) return true;

        for (int col = 0; col < N; col++) {
            if (isSafe(board, row, col)) {
                board[row][col] = 1;
                if (solveNQueensUtil(board, row + 1)) return true;
                board[row][col] = 0;
            }
        }
        return false;
    }

    static boolean isSafe(int[][] board, int row, int col) {
        for (int i = 0; i < row; i++)
            if (board[i][col] == 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 >= 0 && j < N; i--, j++)
            if (board[i][j] == 1) return false;

        return true;
    }

    static void printBoard(int[][] board) {
        for (int[] row : board) {
            for (int cell : row) System.out.print(cell + " ");
            System.out.println();
        }
    }
}
```

```
Enter the size of the board (N): 4
Enter the initial row and column for the first Queen: 0 1
0 1 0 0
0 0 0 1
1 0 0 0
0 0 1 0
```

```
Enter the size of the board (N): 4
Enter the initial row and column for the first Queen: 0 3
No solution exists.
```