Create a function bool SolveKnightsTour(int[,] board, int moveX, int moveY, int moveCount, int[] xMove, int[] yMove) that attempts to solve the Knight's Tour problem using backtracking. The function should return true if a solution exists and false otherwise. The board represents the chessboard, moveX and moveY are the current coordinates of the knight, moveCount is the current move count, and xMove[], yMove[] are the possible next moves for the knight. Fill the chessboard such that the knight visits every square exactly once. Keep the chessboard size to 8x8.

```
public class KnightTourDemo {


        static int N=8;


        static boolean isSafe(int x, int y, int sol[][]) {

                return (x>=0 && x< N && y>= 0 && y<  N && sol[x][y]== -1);

        }


        static void printSolution(int sol[][]) {

                for(int x=0; x<N; x++) {

                        for(int y=0; y<N;y++) {

                                System.out.print(sol[x][y]+ " ");

                        }

                        System.out.println();

                }

        }


                static boolean solveKTUtil(int x, int y, int movei, int sol[][], int
xMove[],int yMove[]) {
```

```
            int k,next_x,next_y;
            if(movei== N*N)
                return true;

            for(k=0; k<8;k++) {
                next_x= x+ xMove[k];
                next_y= y +yMove[k];

                if(isSafe(next_x,next_y,sol)) {
                    sol[next_x][next_y]=movei;

                    if(solveKTUtil(next_x, next_y, movei+1, sol,
xMove,yMove)) {
                        return true;
                }else {
                    sol[next_x][next_y]= -1;
                }
            }
        }
            return false;
    }
    static boolean solveKT() {
        int sol[][]=new int[8][8];

        for(int x=0; x<N; x++)
            for(int y=0; y<N;y++)
                sol[x][y]= -1;
```

```
			int xMove[] = {2,1,-1,-2,-2,-1,1,2};
			int yMove[] = {1,2,2,1,-1,-2,-2,-1};

			sol[0][0]=0;

			if(!solveKTUtil(0,0,1,sol, xMove, yMove)) {
				System.out.println("Solution does not exist");
				return false;
			}else {
				printSolution(sol);
			}
			return true;
		}
			public static void main(String[] args) {
			// TODO Auto-generated method stub
			solveKT();
		}

	}
```