```
//Solve Rat in a Maze problem using backtracking
public class RatinMaze {
	static int N;
	void printSolution(int sol[][])
	{
		for (int i = 0; i < N; i++) {
			for (int j = 0; j < N; j++)
				System.out.print(" " + sol[i][j] + "
");
			System.out.println();
		}
	}
	boolean isSafe(int maze[][], int x, int y)
	{
		return (x >= 0 && x < N && y >= 0 && y < N && maze[x]
[y] == 1);
	}
	boolean solveMaze(int maze[][])
	{
		int sol[][] = new int[N][N];

		if (solveMazeUtil(maze, 0, 0, sol) == false) {
			System.out.print("Solution doesn't exist");
			return false;
		}

		printSolution(sol);
		return true;
	}
	boolean solveMazeUtil(int maze[][], int x, int y, int sol[][])
	{
		if (x == N - 1 && y == N - 1 && maze[x][y] == 1) {
			sol[x][y] = 1;
			return true;
		}
		if (isSafe(maze, x, y) == true) {
			sol[x][y] = 1;
			if (solveMazeUtil(maze, x + 1, y, sol))
				return true;
			if (solveMazeUtil(maze, x, y + 1, sol))
				return true;
			sol[x][y] = 0;
			return false;
		}
		return false;
	}

	public static void main(String args[])
	{
		RatinMaze rat = new RatinMaze();
		int maze[][] = {{1,0,1,0,0},
						 {1,1,1,1,1},
						 {0,1,0,1,0},
						 {1,1,0,1,1},
						 {0,1,1,0,1}};
		N = maze.length;
```

```
                    rat.solveMaze(maze);
          }
}
```