/* Problem 1--Marching Up And Down The Square
This was of course a sliding number puzzle coerced into a Monty
Python skit.
The easy wasy to do this was NOT to envision it as a 4*4 square but
as a one-dimensional array of size 16. */

import java.io.*;
import java.util.*;

public class prob1 {

private static Scanner in;
private static PrintWriter out;
private static int cs, strt;
private static int[] Board;

public static void main (String[] args) throws Exception {
in = new Scanner (new File("prob1.in"));
out = new PrintWriter("prob1.out");
Board = new int[16]; //That single-dimensional array
cs = 1;
while (in.hasNextInt()) { //Read until EOF
   ReadIn ();
   Process ();
}
in.close ();
out.close ();
}

/* All this does is read 16 ints into the Board. */
public static void ReadIn() throws Exception {
   for (int i=0; i < 16; i++) Board[i] = in.nextInt();
   strt = in.nextInt();
}

/* The main thrust of the program.
Here's the thing. Horizontal moves are +/- 1. Vertical moves are
+/- 4. If two positions are the same mod 4, they're in the same
column. If they're the same div 4, they're in the same row. */
public static void Process() throws Exception {
   int sloc = locate (strt), zloc = locate (0); //locate the two
   if (sloc/4 != zloc/4 && sloc%4 != zloc%4) //important points.
      out.printf("Case %d: I can't march up and down the square!\n\n", cs++);
      //They're not aligned vertically or horizontally.
   else {
      int dir;
      if (sloc/4==zloc/4)
         if (sloc < zloc) dir = 1; //Compute the direction to move soldiers.
      else dir = -1;
      else
         if (sloc < zloc) dir = 4;
         else dir = -4;
      for (int x = zloc; x != sloc; x-=dir) Board[x] = Board[x-dir];
      Board[sloc] = 0; //Move the soldiers
      out.printf("Case %d:\n\n",cs++);
      Print();
      out.printf("\n\n");
   }
}

/* The easy wasy to do this was NOT to envision it as a 4*4 square but
as a one-dimensional array of size 16. */

import java.io.*;
import java.util.*;

public class prob1 {

private static Scanner in;
private static PrintWriter out;
private static int cs, strt;
private static int[] Board;

public static void main (String[] args) throws Exception {
in = new Scanner (new File("prob1.in"));
out = new PrintWriter("prob1.out");
Board = new int[16]; //That single-dimensional array
cs = 1;
while (in.hasNextInt()) { //Read until EOF
   ReadIn ();
   Process ();
}
in.close ();
out.close ();
}

/* All this does is read 16 ints into the Board. */
public static void ReadIn() throws Exception {
   for (int i=0; i < 16; i++) Board[i] = in.nextInt();
   strt = in.nextInt();
}

/* The main thrust of the program.
Here's the thing. Horizontal moves are +/- 1. Vertical moves are
+/- 4. If two positions are the same mod 4, they're in the same
column. If they're the same div 4, they're in the same row. */
public static void Process() throws Exception {
   int sloc = locate (strt), zloc = locate (0); //locate the two
   if (sloc/4 != zloc/4 && sloc%4 != zloc%4) //important points.
      out.printf("Case %d: I can't march up and down the square!\n\n", cs++);
      //They're not aligned vertically or horizontally.
   else {
      int dir;
      if (sloc/4==zloc/4)
         if (sloc < zloc) dir = 1; //Compute the direction to move soldiers.
      else dir = -1;
      else
         if (sloc < zloc) dir = 4;
         else dir = -4;
      for (int x = zloc; x != sloc; x-=dir) Board[x] = Board[x-dir];
      Board[sloc] = 0; //Move the soldiers
      out.printf("Case %d:\n\n",cs++);
      Print();
      out.printf("\n\n");
   }
}
/* Find the spot I'm looking for. */
public static int locate (int pos) throws Exception {
    int i=0;
    for (;Board[i]!=pos;i++);
    return i;
}

/* Print the Board */
public static void Print() throws Exception {
    for (int i = 0; i < 4; i++)
        out.printf("%2d %2d %2d %2d
", Board[4*i],Board[4*i+1],Board[4*i+2],Board[4*i+3]);
}