#include<iostream>

#include<iomanip>

#include<cstring>

using namespace std;

int tile=1; //L型骨牌的编号(递增)

int board[100][100];//棋盘

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 递归方式实现棋盘覆盖算法

\* 输入参数：

\* tr--当前棋盘左上角的行号

\* tc--当前棋盘左上角的列号

\* dr--当前特殊方格所在的行号

\* dc--当前特殊方格所在的列号

\* size：当前棋盘的:2^k

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void chessBoard(int tr,int tc,int dr,int dc,int size){

if(size==1)//棋盘方格大小为1,说明递归到最里层

return;

int t=tile++;//每次递增1

int s=size/2;//棋盘中间的行、列号(相等的)

//检查特殊方块是否在左上角子棋盘中

if(dr<tr+s && dc<tc+s)//在

chessBoard(tr,tc,dr,dc,s);

else{//不在，将该子棋盘右下角的方块视为特殊方块

board[tr+s-1][tc+s-1]=t;

chessBoard(tr,tc,tr+s-1,tc+s-1,s);

}

//检查特殊方块是否在右上角子棋盘中

if(dr<tr+s && dc>=tc+s)//在

chessBoard(tr,tc+s,dr,dc,s);

else{//不在，将该子棋盘左下角的方块视为特殊方块

board[tr+s-1][tc+s]=t;

chessBoard(tr,tc+s,tr+s-1,tc+s,s);

}

//检查特殊方块是否在左下角子棋盘中

if(dr>=tr+s && dc<tc+s)//在

chessBoard(tr+s,tc,dr,dc,s);

else{//不在，将该子棋盘右上角的方块视为特殊方块

board[tr+s][tc+s-1]=t;

chessBoard(tr+s,tc,tr+s,tc+s-1,s);

}

//检查特殊方块是否在右下角子棋盘中

if(dr>=tr+s && dc>=tc+s)//在

chessBoard(tr+s,tc+s,dr,dc,s);

else{//不在，将该子棋盘左上角的方块视为特殊方块

board[tr+s][tc+s]=t;

chessBoard(tr+s,tc+s,tr+s,tc+s,s);

}

}

int main(){

while(true){

memset(board,0,sizeof(board));

tile=1;

int size;

cout<<"输入棋盘的size(大小必须是2的n次幂): ";

cin>>size;

int index\_x,index\_y;

cout<<"输入特殊方格位置的坐标: ";

cin>>index\_x>>index\_y;

chessBoard (0,0,index\_x,index\_y,size);

cout<<endl;

for(int i=0;i<size;i++){//输出棋盘覆盖

for(int j=0;j<size;j++ )

cout<<setw(3)<<board[i][j]<<" ";

cout<<endl;

}

cout<<endl;

}

}

