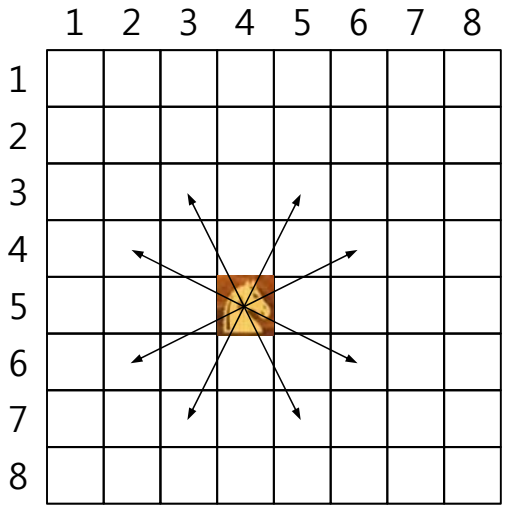
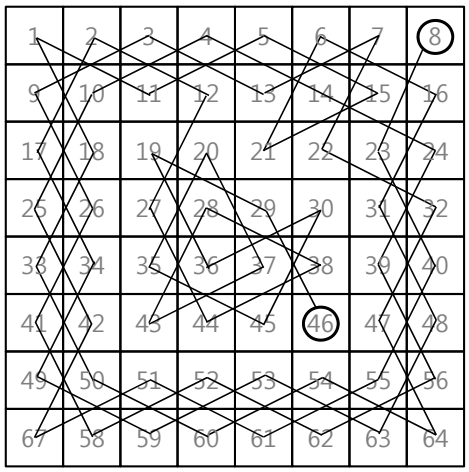
문제

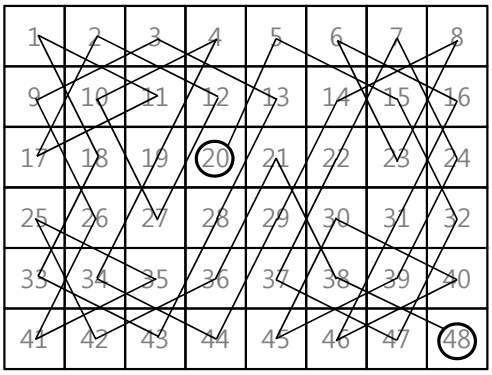
체스판에서 기사(Knight)의 여행

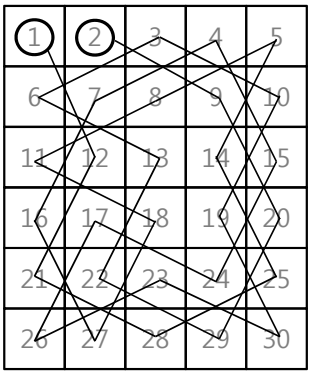
서양 장기인 체스(chess)에는 체스판에서 움직이는 체스말 중에 기사(knight)라는 체스말이 있다. 아래 그림에서와 같이 체스판은 각각 8 개의 행과 열로 만들어진 격자판으로서, 행은 위에서 아래로, 열은 왼쪽에서 오른쪽으로 각각 1 번부터 8 번까지 번호가 부여되어 있다. 또한, 체스판에서 i-번째 행과 j-번째 열에 위치한 작은 정사각형(셀이라 부름)의 위치를 $\langle i, j\rangle$라고 표시하면, 기사는 아래 그림의 체스판에서와 같이 현재의 위치에서 최대 8 가지 방향으로 움직일 수 있다. 즉, 현재 위치가 $\langle i, j\rangle$인 셀에 있는 기사는 다음에는 위치가 $\langle i-2, j-1\rangle$, $\langle i-2, j+1\rangle$, $\langle i-1, j+2\rangle$, $\langle i+1, j+2\rangle$, $\langle i+2, j+1\rangle$, $\langle i+2, j-1\rangle$, $\langle i+1, j-2\rangle$, $\langle i-1, j-2\rangle$인 셀로 옮겨갈 수 있다. 옮겨갈 셀의 위치가 체스판을 벗어나게 되면, 그 셀로는 움직일 수 없다.





위 오른쪽 그림은 <1,8>에 위치한 기사가 이전에 방문한 셀은 다시 방문하지 않으면서, 체스판에 있는 모든 셀을 방문하는 기사의 움직임을 나타내는 경로를 나타낸다. 또한 아래 왼쪽 그림은 행과 열의 크기가 각각 6, 8 인 체스판에서 <3,4>에 위치한 기사가 체스판의 모든 48 개의 정사각형을 모두 방문하는 기사의 경로를 나타내며, 오른쪽 그림은 행과 열의 크기가 각각 6, 5 인 체스판에서 위치가 <1,2>인 셀에서 출발하는 기사의 경로를 나타낸다.





특정한 크기의 체스판에서, 기사가 주어진 위치의 셀에서 출발하여 체스판의 모든 셀을 오직 한 번만 방문하면서, 모든 셀을 모두 다 방문하는 경로를 계산하는 프로그램을 작성하시오.

입력

입력 파일의 이름은 "input.txt" 이다. 입력은 t 개의 테스트 케이스로 주어진다. 입력 파일의 첫 번째 줄에 테스트 케이스의 개수를 나타내는 정수 t 가 주어진다. 두 번째 줄부터 t 개의 줄에는 한 줄에 한 개의 테스트 케이스에 해당하는 네 개의 정수 $m\ n\ s\ t$ ($2 \le m, n \le 8, 1 \le s \le m, 1 \le t \le n$)가 입력된다. 첫 번째 정수 m 은 체스판의 행의 개수를 나타내고, 두 번째 정수 n 은 열의 개수를 나타낸다. 또한 두 정수 s, t 는 각각 기사의 처음 출발한 셀의 위치를 나타내는 행과 열의 번호를 나타낸다. 각 정수들 사이에는 한 개의 공백이 있으며, 잘못된 데이터가 입력되는 경우는 없다.

출력

출력은 표준출력(standard output)을 사용한다. 입력되는 테스트 케이스의 순서대로 다음 줄에 이어서 각 테스트 케이스의 결과를 출력한다. 각 테스트 케이스에 해당하는 출력에는 주어진 체스판에서 기사가 초기 셀에서 출발하여 체스판의 모든 다른 셀을 방문하는 경로가 존재하면 1 을 출력하고 그렇지 않으면 0 을 출력한다. 경로가 존재하는 경우에는 기사가 출발하는 셀에 1 번을 부여하고, 그 다음으로 옮겨가는 셀들에는 순서대로 2, 3, 4, … 등으로 부여하여, 이러한 번호를 첫 번째 행부터 한 줄에 한 행씩 출력한다. 또한 같은 행에서는 첫 번째 열부터 마지막 열까지 순서대로 출력한다. 같은 줄에 출력되는 각 정수들 사이에는 한 개의 공백을 둔다.

입력과 출력의 예

입력	출력
3 6 8 3 4 6 5 1 2 4 4 1 2	1 32 29 38 25 2 17 8 19 39 26 31 28 37 20 3 16 30 33 24 1 46 7 18 9 43 40 27 36 21 12 15 4 34 23 42 45 6 47 10 13 41 44 35 22 11 14 5 48 1 30 1 8 17 24 7 18 23 2 9 22 29 6 25 16 19 14 21 10 3 28 5 12 15 26 13 20 27 4 11 0

(참고) 위 첫 번째와 두 번째 입력 테스트 데이터의 경우에 있어서, 모든 셀을 방문하는 기사의 경로는 위의 출력에서 제시된 경로 이외에도 다른 많은 경로가 존재한다. 채점 프로그램에서는 위에서 제시된 경로 이외에도 다른 가능한 여러 경로 중에서 어느 한 가지가 출력되더라도 정답으로 채점한다.

KnightTour.h

```
#ifndef _KNIGHT_TOUR_H_
#define _KNIGHT_TOUR_H_
#include <iostream>

using namespace std;

const int MAX_SIZE = 9;

class KnightTour
{
public:
    KnightTour();
    KnightTour(int sizeRow, int sizeCol);

    bool buildKnightTour(int startRow, int startCol);
    void printBoard();

private:
    typedef int boardType[MAX_SIZE][MAX_SIZE];

    bool recurKnightTour(int startRow, int startCol,int move);
    bool isValidMove (int row, int col);

    int sizeRow, sizeCol;
    boardType board;
};

#endif // _KNIGHT_TOUR_H_
```

KnightTour.cpp

```
#include "KnightTour.h"

static int direction[8][2] = {{1, -2}, {2, -1}, {2, 1}, {1, 2},
                              {-1, 2}, {-2, 1}, {-2, -1}, {-1, -2}};

KnightTour::KnightTour()
{
    sizeRow = sizeCol = 8;  // default size of Chessboard
}

KnightTour::KnightTour(int sRow, int sCol)
{
    if (sRow>=2 && sRow<=MAX_SIZE)
        sizeRow = sRow;
    else
        sizeRow = 8;    // set to default size

    if (sCol>=2 && sCol<=MAX_SIZE)
        sizeCol = sCol;
    else
        sizeCol = 8;    // set to default size
}

void KnightTour::printBoard()
{
    for (int r = 0; r < sizeRow; r++)
    {
        for (int c = 0; c < sizeCol; c++)
            cout << board[r][c] << " ";

        cout <<endl;
    } // for r
}
```

```
bool KnightTour::buildKnightTour(int startRow, int startCol)
{
    for (int r=0; r<sizeRow; r++)
        for (int c=0; c<sizeCol; c++)
            board[r][c]= 0;

    if (startRow <= 0 || startRow > sizeRow)
        startRow = 0;
    if (startCol <= 0 || startCol > sizeCol)
        startCol = 0;

    board[startRow-1][startCol-1] = 1;

    return recurKnightTour(startRow-1, startCol-1, 1);
}

bool KnightTour::recurKnightTour(int row, int col, int move)
{
    int nextRow, nextCol;

    if (move == sizeRow*sizeCol)
        return true;

    for(int dir=0; dir<8; dir++)
    {
        nextRow = row + direction[dir][0];
        nextCol = col + direction[dir][1];

        if(isValidMove(nextRow, nextCol))
        {

         ..........................................................

         ..........................................................

         ..........................................................

        }
    }

    return false;
}

bool KnightTour::isValidMove(int row, int col)
{
    if(row>=sizeRow || row<0)
        return false;

    if(col>=sizeCol || col<0)
        return false;

    if(board[row][col] != 0)
        return false;

    else
        return true;
}
```

TestKnightTour.cpp

```
#include <fstream>
#include <cstdlib>
#include "KnightTour.h"


void main()
```

```
{
    ifstream inStream;
    int numTestCases;

    inStream.open("input.txt");
    if (inStream.fail())
    {
        cerr << "Input file opening failed.\n";
        exit(1);
    }

    inStream >> numTestCases;

    for(int i=0; i<numTestCases; i++)
    {
        int sizeRow, sizeCol;
        int startRow, startCol;

        inStream >> sizeRow >> sizeCol;
        inStream >> startRow >> startCol;

        KnightTour kTour(sizeRow, sizeCol);

        if (kTour.buildKnightTour(startRow, startCol))
        {
            cout << 1 << endl;
            kTour.printBoard();
        }
        else
            cout << 0 << endl;
    }

    inStream.close();
}
```