1219：马走日

题目描述

马在中国象棋以日字形规则移动。

请编写一段程序，给定n×m大小的棋盘，以及马的初始位置(x，y)，要求不能重复经过棋盘上的同一个点，计算马可以有多少途径遍历棋盘上的所有点。

输入格式

第一行为整数T(T < 10)，表示测试数据组数。

每一组测试数据包含一行，为四个整数，分别为棋盘的大小以及初始位置坐标n,m,x,y。(0≤x≤n-1,0≤y≤m-1，m < 10，n < 10)。

输出格式

每组测试数据包含一行，为一个整数，表示马能遍历棋盘的途径总数，0为无法遍历一次。

输入样例

```
1
5 4 0 0
```

输出样例

```
32
```

解析

关键字“途径总数”，因此，深搜是不二之选，直接上模板。

编码

```
#include <bits/stdc++.h>

using namespace std;

//1、终止条件是什么？
//走过的总点数等于棋盘的总点数

//八个移动方向
int const FORWARD_NUM = 8;

int Count = 0;    //总的方案数

int n, m;  //行列数
```

```
//八个移动方向
int Forward[8][2] = { //八方向，二维（行，列）
       {-2, -1},//左上1
         {-1, -2},//左上2
         {1,  -2},//左下1
         {2,  -1},//左下2

         {-2, 1},//右上1
         {-1, 2},//右上2
         {1,  2},//右下1
         {2,  1},//右下2
};

//记录当前节点是否行走过
bool Vis[11][11];

//判断指定的目标点是否可以行走
bool Check(int x, int y) {
    //1、是否越界
    if (x >= 0 && y >= 0 && x < n && y < m) {
        //2、未访问过
        if (!Vis[x][y]) {
            return true;
        }
    }
    return false;
}

//深搜代码
void Dfs(int sx, int sy, int step) {
    //更新当前总的移动步数
    //当遍历完毕全部的棋盘时
    if (step == n * m) {
        //总方案数加1，并停止搜索;
        Count++;
        return;
    }

    for (int i = 0; i < FORWARD_NUM; ++i) {
        //新的xy坐标
        int newX = sx + Forward[i][0];
        int newY = sy + Forward[i][1];
        if (Check(newX, newY)) {
            //标记为已经行走
          Vis[newX][newY] = true;
            //执行深搜
          Dfs(newX, newY, step + 1);
            //回溯
```

```
            Vis[newX][newY] = false;
        }
    }
}

//重置
void Reset() {
    Count = 0;
    memset(Vis, false, sizeof(Vis));
}

void Read() {
    int T;  //测试组数
   cin >> T;
    int sx, sy;
    for (int i = 0; i < T; ++i) {
        cin >> n >> m >> sx >> sy;
        Reset();
        Vis[sx][sy] = true;
        Dfs(sx, sy, 1);
        cout << Count << endl;
    }
}


int main() {
    Read();
    return 0;
}
```