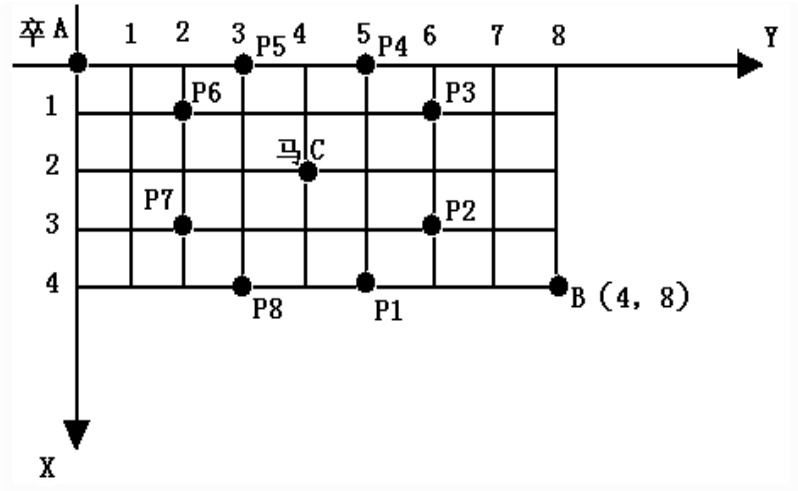
1314：过河卒(Noip2002)

题目描述

棋盘上A点有一个过河卒，需要走到目标B点。卒行走的规则：可以向下、或者向右。同时在棋盘上的某一点有一个对方的马（如C点），该马所在的点和所有跳跃一步可达的点称为对方马的控制点，如图3-1中的C点和P1，……，P8，卒不能通过对方马的控制点。棋盘用坐标表示，A点(0,0)、B点(n，m) (n,m为不超过20的整数),同样马的位置坐标是需要给出的，C≠A且C≠B。现在要求你计算出卒从A点能够到达B点的路径的条数。



输入

给出n、m和C点的坐标。

输出

从A点能够到达B点的路径的条数。

输入样例

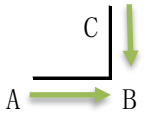
8 6 0 4

输出样例

1617

解析

这是一道非常简单的递推问题，我们很容易想到，能够到达B点的路线，只能来源于上方和左侧。如下图所示：



因此，我们只需要将能够到达左侧(A点)和上方(C点)的方案数加起来就可以了，怎么样，是不是非常类似上楼梯？进一步我们又会发现，A点和C点的方案数也是分别来自各自的上方和左侧，以此类推，很容易得到递推公式，我们用数组a[x][y]代表到达点x,y的方案数，则有：

a[x][y] = a[x-1][y] + a[x][y-1]

其中x代表列数，y代表行数

需要注意的是，其中有一些点可能在范围以外，或者是马的控制点，这些点对于我们来说都是不可行走的，因此，这些点的方案数都为0。

小提示：一定要确保x,y的含义正确。

编码

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

using namespace std;

//马的相对偏移坐标
int offset[9][2] = {{0,  0},
                    {1,  2},
                    {1,  -2},
                    {-1, 2},
                    {-1, -2},
                    {2,  1},
                    {2,  -1},
                    {-2, 1},
                    {-2, -1}};

//障碍点数组
bool block[22][22];
//大数记录，防止超时
long long record[22][22];

int Bx, By, horseX, horseY;

void InitBlock() {
    for (int i = 0; i < 9; i++) {
        int tempx = horseX + offset[i][0];
        int tempy = horseY + offset[i][1];
        //判断障碍点是否越界
       if (tempx >= 0 && tempx <= Bx && tempy >= 0 && tempy <= By) {
            block[tempx][tempy] = true;
        }
    }
}

long long FindPath(int x, int y) {
    //从缓存中返回数据
   if (record[x][y] != 0) {
        return record[x][y];
```

```
    }
    //当前位障碍点 或者非法点
    if (block[x][y] || x < 0 || y < 0) {
        return 0;
    }
    //到达起点
    if (x == 0 && y == 0) {
        return 1;
    }

    long long res1 = 0, res2 = 0;
    //计算左边的点有多少方案
    if (x > 0) {
        res1 = FindPath(x - 1, y);
    }
    //计算上边的点有多少方案
    if (y > 0) {
        res2 = FindPath(x, y - 1);
    }

    long long res = res1 + res2;

    return record[x][y] = res;
}

int main(int argc, char **argv) {
    cin >> Bx >> By >> horseX >> horseY;
    InitBlock();
    cout << FindPath(Bx, By);
    return 0;
}
```

逻辑航线培优教育，信息学奥赛培训专家。

扫码添加作者获取更多内容。