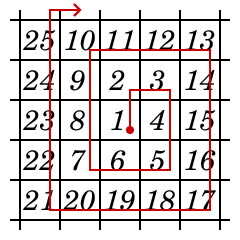
**算法训练 Entertaining Geodetics**

时间限制：2.0s   内存限制：256.0MB

问题描述

　　在此游戏中地图被分为了许多叫作Geo格的正方形方格，其中一些被涂上色，假设没有涂色的为透明色。  
　　地图中还有些Geo符号，它们样子像不同颜色的金字塔（包括透明Geo符号）。每个Geo符号都坐落在Geo格上，每个Geo格上最多一个Geo符号。  
　　Geo符号可以被消除。为了更好地理解Geo符号在消除时发生了什么，这里引入把刚消除的Geo符号放入的队列。  
　　从队列中取出Geo符号，观察包含Geo符号的Geo格的颜色，如果它不是透明的且颜色不同于Geo符号，则把所有这个颜色的Geo格重新涂为Geo符号的颜色（透明的Geo符号则为透明色）。重涂色是在一个无限大的区域从那个有符号的Geo格子开始螺旋状进行的。  
　　.  
　　换句话说，我们选择所有需要重涂色的方格找到它们在以有符号格为中心的无限螺旋表格中所对应的数字。此后按数字的增加顺序我们对其重染色。  
　　如果在重染色时遇到一个格子包含另一个Geo符号的情况则将Geo符号移出并放置在队列尾部。  
　　当重染色结束后Geo符号彻底消失，并且队列中下一个Geo符号（如果有）将取出，重复此操作直至队列为空。  
　　为了更好地理解请看一个例子。  
　　你知道所有格子的颜色、所有符号的位置。计算出队列里符号彻底消失时所造成的重染色次数。  
　　推荐使用I64d输出。

输入格式

　　第一行包含两个数n，m(1<=n，m<=300)—地图的高和宽。  
　　接下来n行每行m个数—格子的颜色。  
　　接下来n行每行m个数—对符号的描述，-1表示没有符号，否则数字代表符号的颜色。  
　　所有颜色都是属于0到10^9的整数，0表示透明。  
　　最后一行两个数x，y(1<=x<=n，1<=y<=m)—需要消除的Geo符号的行和列位置。行从上到下标记，列从左往右标记，从1开始。保证位置(x，y)包含一个符号。

输出格式

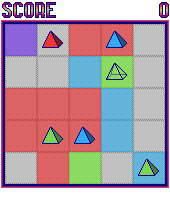
　　一行一个数—符号消除时重染色次数。

样例输入

5 5  
9 0 1 1 0  
0 0 3 2 0  
1 1 1 3 0  
1 1 1 3 0  
0 1 2 0 3  
-1 1 -1 3 -1  
-1 -1 -1 0 -1  
-1 -1 -1 -1 -1  
-1 2 3 -1 -1  
-1 -1 -1 -1 2  
4 2

样例输出

35

样例说明

C++:

#include<cstring>

#include<stdio.h>

#include<vector>

#include<map>

#include<queue>

#include<algorithm>

#define mp make\_pair

#define pb push\_back

#define fi first

#define se second

#define fo(i,a,b) for(i=a;i<=b;++i)

using namespace std;

typedef long long LL;

typedef pair<int,int> node;

const int mn=310,mm=mn\*mn;

int a[mn][mn],b[mn][mn],spi[mn\*2][mn\*2];

int n,m,l,tt,color;

map<int,int>ha;

vector<node>e[mm];

node w[mm];

queue<node>Q;

LL ans=0;

int hash(int p)

{

if(ha.count(p))return ha[p];

return ha[p]=++color;

}

void init\_spiral()

{

int tt=0,x,y,i;

l=max(n,m);

spi[x=l][y=l]=++tt;

for(int L=2,ll=2;ll<=l;L+=2,++ll)

{

--x,--y;

fo(i,1,L)spi[x][++y]=++tt;

fo(i,1,L)spi[++x][y]=++tt;

fo(i,1,L)spi[x][--y]=++tt;

fo(i,1,L)spi[--x][y]=++tt;

}

}

int calc(int x1,int y1,int x2,int y2)

{

int x=x2-x1+l,y=y2-y1+l;

return spi[x][y];

}

int main()

{

scanf("%d%d",&n,&m);

int i,j;

fo(i,1,n)

fo(j,1,m)

{

scanf("%d",a[i]+j);

if(a[i][j])

a[i][j]=hash(a[i][j]);

e[a[i][j]].pb(mp(i,j));

}

fo(i,1,n)

fo(j,1,m)

{

scanf("%d",b[i]+j);

if(b[i][j]>0)b[i][j]=hash(b[i][j]);

}

init\_spiral();

int x,y,nx,ny,c,s=0;

node p;

scanf("%d%d",&x,&y);

Q.push(mp(x,y)),c=a[x][y];

while(!Q.empty())

{

p=Q.front(),Q.pop();

x=p.fi,y=p.se;

if(b[x][y]==c)continue;

if(!c)break;

tt=0;

for(i=0;i<e[c].size();++i)

{

nx=e[c][i].fi,ny=e[c][i].se;

if(b[nx][ny]!=-1&&(nx!=x||ny!=y))

w[++tt]=mp(calc(x,y,nx,ny),i);

++s;

}

sort(w+1,w+1+tt);

fo(i,1,tt)Q.push(e[c][w[i].se]);

ans+=s,e[c].clear(),c=b[x][y];

}

printf("%I64d\n",ans);

return 0;

}