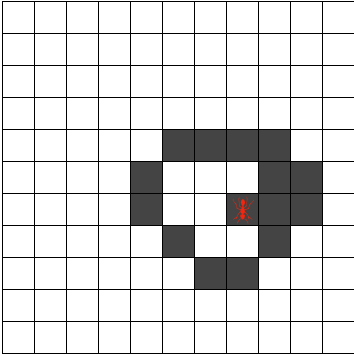
**兰顿蚂蚁**

问题描述

  
  
　　兰顿蚂蚁，是于1986年，由克里斯·兰顿提出来的，属于细胞自动机的一种。  
  
　　平面上的正方形格子被填上黑色或白色。在其中一格正方形内有一只“蚂蚁”。  
　　蚂蚁的头部朝向为：上下左右其中一方。  
  
　　蚂蚁的移动规则十分简单：  
　　若蚂蚁在黑格，右转90度，将该格改为白格，并向前移一格；  
　　若蚂蚁在白格，左转90度，将该格改为黑格，并向前移一格。  
  
　　规则虽然简单，蚂蚁的行为却十分复杂。刚刚开始时留下的路线都会有接近对称，像是会重复，但不论起始状态如何，蚂蚁经过漫长的混乱活动后，会开辟出一条规则的“高速公路”。  
  
　　蚂蚁的路线是很难事先预测的。  
  
　　你的任务是根据初始状态，用计算机模拟兰顿蚂蚁在第n步行走后所处的位置。

输入格式

　　输入数据的第一行是 m n 两个整数（3 < m, n < 100），表示正方形格子的行数和列数。  
　　接下来是 m 行数据。  
　　每行数据为 n 个被空格分开的数字。0 表示白格，1 表示黑格。  
  
　　接下来是一行数据：x y s k, 其中x y为整数，表示蚂蚁所在行号和列号（行号从上到下增长，列号从左到右增长，都是从0开始编号）。s 是一个大写字母，表示蚂蚁头的朝向，我们约定：上下左右分别用：UDLR表示。k 表示蚂蚁走的步数。

输出格式

　　输出数据为两个空格分开的整数 p q, 分别表示蚂蚁在k步后，所处格子的行号和列号。

样例输入

5 6  
0 0 0 0 0 0  
0 0 0 0 0 0  
0 0 1 0 0 0  
0 0 0 0 0 0  
0 0 0 0 0 0  
2 3 L 5

样例输出

1 3

样例输入

3 3  
0 0 0  
1 1 1  
1 1 1  
1 1 U 6

样例输出

0 0

本题的C++参考代码如下：

//天农计算机许晓华老师出品

#include<iostream>

using namespace std;

#define N 100

int dx[]={-1,0,1,0};

int dy[]={0,1,0,-1};

char d[]={'U','R','D','L'};//上右下左

int main()

{

int a[N][N],m,n,i,j;

cin>>m>>n;

for(i=0;i<m;i++)

for(j=0;j<n;j++)

cin>>a[i][j];

int x,y,k;

char s;

cin>>x>>y>>s>>k;

for(i=0;i<4;i++)

{

if(d[i]==s) break;

}

while(k--)

{

if(a[x][y])//黑格

i=(i+1)%4;//右转

else//白格

i=(i+3)%4;//左转

a[x][y]^=1;//黑白互换

//if(x>0&&y>0&&x<m-1&&y<n-1)

if(!x&&!i) continue;//不能上

if(y==n-1&&i==1) continue;//不能右

if(x==m-1&&i==2) continue;//不能下

if(!y&&i==3) continue;//不能左

x=x+dx[i],y=y+dy[i];

}

cout<<x<<" "<<y;

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

int a[1000][1000]={

0

};

char f(int i,char now)

{

if(i==0&&now=='U'||i==1&&now=='D')

return 'R';

else if(i==0&&now=='R'||i==1&&now=='L')

return 'D';

else if(i==0&&now=='D'||i==1&&now=='U')

return 'L';

else if(i==0&&now=='L'||i==1&&now=='R')

return 'U';

}

int main()

{

int n,m,i1,i2,x,y,k;

char s;

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

for(i1=0;i1<n;i1++)

for(i2=0;i2<m;i2++)

scanf("%d",&a[i1][i2]);

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

getchar();

scanf("%c%d",&s,&k);

char now=s;

while(k--)

{ if(a[x][y]==0)

a[x][y]=1;

else

a[x][y]=0;

now=f(a[x][y],now);

if(now=='U')

x--;

else if(now=='D')

x++;

else if(now=='L')

y--;

else

y++;

}

printf("%d %d\n",x,y);

return 0;

}

本题的Java参考代码如下：

import java.io.\*;

public class Main {

static int n, m;

static int s, e;

static char[] chs = { 'L', 'U', 'R', 'D', 'L' };

static int count = 0;

public static void main(String[] args) throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

String s1[] = br.readLine().split(" ");

n = Integer.parseInt(s1[0]);

m = Integer.parseInt(s1[1]);

int[][] arr = new int[n][m];

for (int a = 0; a < n; a++) {

String str[] = br.readLine().split(" ");

for (int b = 0; b < m; b++) {

arr[a][b] = Integer.parseInt(str[b]);

}

}

int x, y;

String s2[] = br.readLine().split(" ");

x = Integer.parseInt(s2[0]);

y = Integer.parseInt(s2[1]);

char dec = s2[2].charAt(0);

int z;

z = Integer.parseInt(s2[3]);

s = x;

e = y;

while (count < z) {

if (arr[s][e] == 1) {

for (int i = 0; i < chs.length; i++) {

if (dec == chs[i]) {

dec = chs[i + 1];

break;

}

}

arr[s][e] = 0;

func(dec, s, e);

}

if (arr[s][e] == 0) {

for (int j = 1; j < chs.length; j++) {

if (dec == chs[j]) {

dec = chs[j - 1];

break;

}

}

arr[s][e] = 1;

func(dec, s, e);

}

}

System.out.print(s + " " + e);

}

public static void func(char dec, int x, int y) {

if (dec == 'L') {

e -= 1;

count++;

}

if (dec == 'U') {

s -= 1;

count++;

}

if (dec == 'R') {

e += 1;

count++;

}

if (dec == 'D') {

s += 1;

count++;

}

}

}