牧场危机

一. 考察内容：

动态规划 路径记录

二. 题目分析：

[题目大意]

一个棋盘上有一些奶牛群和草堆，每个奶牛群访问一次草堆，答案加一，你有k次机会可以控制所有奶牛向某个方向移动一格，求最大答案和该答案的最小字典序方案。

[写题思路]

很显然，我们预处理出奶牛群偏移量为(i,j)时能获得的贡献，设f[i][j][k]为前i次控制之后，偏移量为(j,k)的答案，转移就是从四个方向转移即可。

记录方案是一个很麻烦的事情，我们需要开一个四维数组road[i][j][k][l]表示状态f[i][j][k]时最大答案的最小字典序中第l个字母，对于每次转移，找到所有转移方案中原先字典序最小的方案优先转移。

三. 代码实现：

```
#define _CRT_SECURE_NO_DEPRECATE
/*************************
*创建时间：2018 09 13
*文件类型：源代码文件
*题目来源：BZOJ
*当前状态：已通过
*备忘录：动态规划
*作者：HtBest
*************************/
#include <stdio.h>
#include <iostream>
#include <string>
#include <stdlib.h>
#include <string.h>
#include <algorithm>
#include <queue>
#include <bitset>
// #include <sys/wait.h>
// #include <sys/types.h>
// #include <unistd.h>
using namespace std;
#define MAXN 1001
#define MAXK 31
char dic[5]="ENSW";
int n,m,K,a[90][90],f[MAXK][90][90],map[MAXN][MAXN],road[MAXK][90][90][MAXK];
struct NODE
{
    int x,y;
    NODE(int x=0,int y=0):x(x),y(y){}
}cow[MAXN];
/* Variable explain:
n:牛数
m:草堆数
K:口令数
a[i][j]:奶牛向下移动i-50，向右移动j-50格时能踩到多少草堆
f[i][j][k]:前i个口令之后，奶牛偏移(j-50,K-50)时的最大答案
map[i][j]:地图
cow[i]:第i个奶牛的位置
road[i][j][k]:f对应的方案
*/
```

```
void read()
{
    int ls1,ls2;
    scanf("%d%d%d",&n,&m,&K);
    for(int i=1;i<=n;++i)scanf("%d%d",&ls1,&ls2),cow[i]=NODE(ls1,ls2);
    for(int i=1;i<=m;++i)scanf("%d%d",&ls1,&ls2),map[ls1][ls2]=1;
    return;
}
void test()
{
    for(int i=1;i<=100;++i)
    {
        for(int j=1;j<=100;++j)
        {
            int flag=0;
            for(int k=1;k<=n;++k)
            {
                if(cow[k].x==i&&cow[k].y==j){flag=1;break;}
            }
            if(map[i][j])printf("* ");
            else printf("%c ",flag?'#':'.');
        }
        puts("");
    }
    for(int i=1;i<=30;++i)printf("-");puts("");

    for(int i=-K;i<=K;++i)
    {
        for(int j=-K;j<=K;++j)printf("%d ",a[i+50][j+50]);
        puts("");
    }
}
void init()
{
    for(int i=-K;i<=K;++i)
    {
        for(int j=-K;j<=K;++j)
        {
            for(int k=1;k<=n;++k)
            {
                if(cow[k].x+i<=0||cow[k].x+i>1000||cow[k].y+j<=0||
cow[k].y+j>1000)continue;
                if(map[cow[k].x+i][cow[k].y+j])a[i+50][j+50]++;
            }
        }
    }
    // test();
}
bool cmp(int a[],int b[],int x)
{
    for(int i=1;i<=x;++i)if(a[i]<b[i])return true;else if(a[i]>b[i])return false;
    return true;
}
void dp()
{
    int ans=0,ansi[1000],ansj[1000],_ans=0,realansi=1,realansj=1;
    for(int i=0;i<=K;++i)for(int j=1;j<=100;++j)for(int k=1;k<=100;++k)f[i][j][k]=-1e9;
    for(int i=0;i<=K;++i)for(int j=1;j<=100;++j)for(int k=1;k<=100;++k)for(int
l=1;l<=30;++l)road[i][j][k][l]=0;
    f[0][50][50]=0;
    for(int i=1;i<=K;++i)
    {
        for(int j=-i;j<=i;++j)
        {
            for(int k=-abs(i-abs(j));k<=abs(i-abs(j));++k)
            {
                int I=j+50,J=k+50;
                int oldf=max(max(f[i-1][I-1][J],f[i-1][I+1][J]),max(f[i-1][I]
[J-1],f[i-1][I][J+1]));
                if(oldf==f[i-1][I-1][J])
                {
                    if(road[i][I][J][1]==0||cmp(road[i-1][I-1][J],road[i][I][J],i-1))
```

```
                    {
                        for(int l=1;l<i;++l)road[i][I][J][l]=road[i-1][I-1][J][l];
                        road[i][I][J][i]=0;
                    }
                }
                if(oldf==f[i-1][I][J-1])
                {
                    if(road[i][I][J][1]==0||cmp(road[i-1][I][J-1],road[i][I][J],i-1))
                    {
                        for(int l=1;l<i;++l)road[i][I][J][l]=road[i-1][I][J-1][l];
                        road[i][I][J][i]=1;
                    }
                }
                if(oldf==f[i-1][I][J+1])
                {
                    if(road[i][I][J][1]==0||cmp(road[i-1][I][J+1],road[i][I][J],i-1))
                    {
                        for(int l=1;l<i;++l)road[i][I][J][l]=road[i-1][I][J+1][l];
                        road[i][I][J][i]=2;
                    }
                }
                if(oldf==f[i-1][I+1][J])
                {
                    if(road[i][I][J][1]==0||cmp(road[i-1][I+1][J],road[i][I][J],i-1))
                    {
                        for(int l=1;l<i;++l)road[i][I][J][l]=road[i-1][I+1][J][l];
                        road[i][I][J][i]=3;
                    }
                }
                f[i][I][J]=oldf+a[I][J];
                if(ans<f[i][I][J])ans=f[i][I][J],ansi[_ans=1]=I,ansj[_ans]=J;
                else if(ans==f[i][I][J])ansi[++_ans]=I,ansj[_ans]=J;
                // if(i<=2)printf("f[%d][%d][%d]=%d\n",i,j,k,f[i][I][J]);
            }
        }
    }
    realansi=ansi[1],realansj=ansj[1];
    for(int i=1;i<=_ans;++i)
    {
        if(cmp(road[K][ansi[i]][ansj[i]],road[K][realansi][realansj],K))
            realansi=ansi[i],realansj=ansj[i];
    }
    printf("%d\n",ans);
    for(int i=1;i<=K;++i)printf("%c",dic[road[K][realansi][realansj][i]]);
    puts("");
}
int main()
{
    // freopen(".in","r",stdin);
    // freopen(".out","w",stdout);
    read();
    init();
    dp();
    return 0;
}
```

<题目跳转> <查看代码>