100%：

1) 用曼哈顿距离作为下标，建立一棵线段树。

对于每一只鸟，我们确定了它的活动范围后

update(最短曼哈顿距离，最长曼哈顿距离，1，maxm，1)

更新过程时间复杂度 O(nlogn)

对于每一次的查询，我们可以看做是单点查询，

query(查询的点，1，maxm,1)

查询过程时间复杂度 O(Q logn)

总时间复杂度(nlogn)

2) 差分：

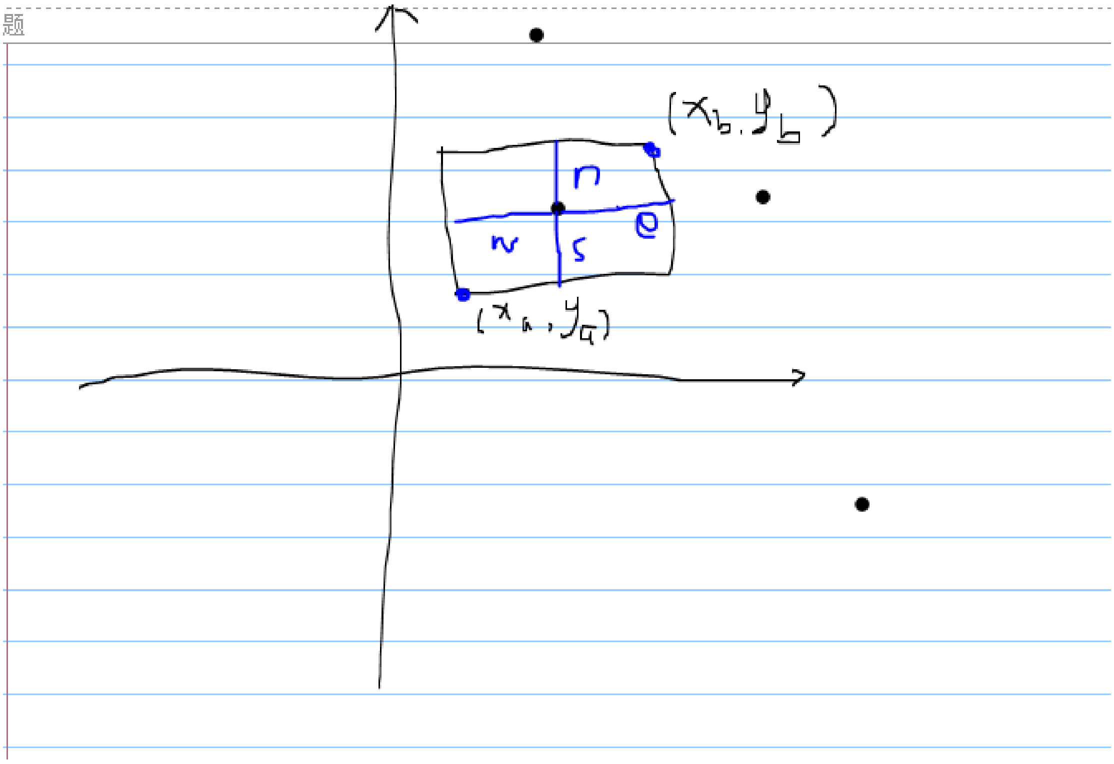
对于每一只鸟，我们确定了它的活动范围后

change[最短曼哈顿距离]+1，change[最大曼哈顿距离+1]-1

扫描求前缀和

总时间复杂度 O(n)

浅谈小鸟活动区域到监控器的曼哈顿距离求法：



(由于题目中没有说 xa<xb&&ya<yb，所以如果 xa>xb||ya>yb,我们要调整 xa,xb;ya,yb 的顺序)

首先假设这个点刚好就落在我们小鸟的活动区域内

记这个点到矩形四条边的距离分别为 w, s, e, n

四个顶点的距离到监控器的曼哈顿距离就可以求出来了，

而最大曼哈顿距离就出现在这四个顶点上

最短曼哈顿距离需要我们分类讨论

如图

具体写法如下：

```
void calc(int xa,int xb,int ya,int yb){
    if(xa>xb) swap(xa,xb);
    if(ya>yb) swap(ya,yb);
    int dis1,dis2,dis3,dis4;
    int e,s,w,n,maxm,minm;
    w=mabs(x-xa),e=mabs(xb-x),s=mabs(yb-y),n=mabs(y-ya);
    dis1=w+n,dis2=e+n,dis3=w+s,dis4=e+s;
    maxm=max(max(dis1,dis2),max(dis3,dis4));
    minm=min(min(dis1,dis2),min(dis3,dis4));
    if(x>=xa&&x<=xb){
        if(y>=ya&&y<=yb) minm=0;
        else minm=min(n,s);
    }
    else if(y>=ya&&y<=yb){
        minm=min(e,w);
    }
    update(minm,maxm,0,maxd,1,1);
}
```