[报告] B

[source] http://202.114.18.202:8080/judge/contest/view.action?cid=6199#problem/B

[Description]

在一个 N*N 的矩阵中，找每一个边长为 2*r+1 的矩阵的中位数。

[方法]

首先可以求出左上角(2*r+1)*(2*r+1)矩阵的中位数，然后将这个矩阵每次移动一个位置，都可以只删除 2*r+1 个元素并添加 2*r+1 个新元素完成下一个矩阵中位数。然后将原问题化为动态求中位数的问题。也就化为求第 K 大数的问题。我们这里可以用树状数组和线段树或者平衡树来达到 logn 求第 K 大数。注意此题用线段树的话会 T 掉，然后我用的是二分加树状数组，虽然复杂度比线段树多了一个 logn，但是速度比线段树要快的多。

另外这个是计算机图形学中的经典问题，据戴牛称：“据 yayamao 大神说 photoshop 源码里有 n^2logn 的算法，另外某论文有 n^2 的算法 ”

[Code]

```
#include<iostream>
#include<cstring>
#include<cstdio>
using namespace std;
#define N 510
#define M 1111222
```

```
int lowbit(int k)
{
	return k&-k;
}
const int maxm=1000003,maxn=maxm*2+10;
int tree[maxm];
int ll;
void change(int k,int d)
{
	while (k<=ll)
	{
		tree[k]+=d;
		k+=lowbit(k);
	}
}
int ask(int k)
{
	int l=0,r=ll;
	while (l+1<r)
	{
		int mid=(l+r)/2;
		int s=0;
```

```
            while (mid)
            {
              s+=tree[mid];
              mid-=lowbit(mid);
            }
            if (s<k) l=(l+r)/2;else r=(l+r)/2;
        }
        return r;
}
int a[510][510];
int ans[510][510];

int main(void)
{
        int n,r;
        while(scanf("%d%d",&n,&r)&&(n||r))
        {
                int ma=0;
                memset(tree,0,sizeof(tree));
                for(int i=1;i<=n;i++)
                {
                        for(int j=1;j<=n;j++)
```

```
        {
            scanf("%d",&a[i][j]);
            a[i][j]++;
            ma=max(ma,a[i][j]);
        }
}
ll=ma;
for(int i=1;i<=2*r+1;i++)
{
    for(int j=1;j<=2*r+1;j++)
    {
        change(a[i][j],1);
    }
}
ans[1][1]=ask(2*r*r+2*r+1)-1;
for(int i=1;i<=n-2*r;i++)
{
    if(i%2==1)
    {
        for(int j=1;j<=n-2*r;j++)
        {
            if(i==1&&j==1) continue;
```

```
                if(j!=1)
                {
                    for(int k=1;k<=2*r+1;k++)
                    {
                        change(a[i-1+k][j-1],-1);
                        change(a[i-1+k][j+2*r],1);
                    }
                    ans[i][j]=ask(2*r*r+2*r+1)-1;
                }
                else
                {
                    for(int k=1;k<=2*r+1;k++)
                    {
                        change(a[i-1][j-1+k],-1);
                        change(a[i+2*r][j-1+k],1);
                    }
                    ans[i][j]=ask(2*r*r+2*r+1)-1;
                }
            }
        }
        else
        {
```

```
        for(int j=n-2*r;j>=1;j--)
        {
            if(j!=n-2*r)
            {
                for(int k=1;k<=2*r+1;k++)
                {
                    change(a[i-1+k][j+2*r+1],-1);
                    change(a[i-1+k][j],1);
                }
                ans[i][j]=ask(2*r*r+2*r+1)-1;
            }
            else
            {
                for(int k=1;k<=2*r+1;k++)
                {
                    change(a[i-1][j-1+k],-1);
                    change(a[i+2*r][j-1+k],1);
                }
                ans[i][j]=ask(2*r*r+2*r+1)-1;
            }
        }
    }
```

```
            }
            for(int i=1;i<=n-2*r;i++)
            {
                for(int j=1;j<=n-2*r;j++)
                {
                    printf("%d ",ans[i][j]);
                }
                printf("\n");
            }
        }
        return 0;
}
```