FFT

补充板子

```
#include<iostream>
#include<cstdio>
#include<cmath>
#include <string.h>
using namespace std;
const int MAXN=4000100;
const double Pi=acos(-1.0);
struct complex {
    double x,y;
    complex (double xx=0,double yy=0) {
        x=xx,y=yy;
    }
} a[MAXN],b[MAXN];
complex operator + (complex a,complex b) {
    return complex(a.x+b.x , a.y+b.y);
}
complex operator - (complex a,complex b) {
    return complex(a.x-b.x , a.y-b.y);
}
complex operator * (complex a,complex b) {
    return complex(a.x*b.x-a.y*b.y , a.x*b.y+a.y*b.x);    //不懂的看复数的运算那部分
}
long long l,r[MAXN];
long long limit=1;
void fast_fast_tle(complex *A,int type) {
    for(int i=0; i<limit; i++)
        if(i<r[i]) swap(A[i],A[r[i]]);//求出要迭代的序列
    for(int mid=1; mid<limit; mid<<=1) { //待合并区间的中点
        complex Wn( cos(Pi/mid) , type*sin(Pi/mid) ); //单位根
        for(int R=mid<<1,j=0; j<limit; j+=R) { //R是区间的右端点□j表示前已经到哪个位置了
            complex w(1,0);//幂
            for(int k=0; k<mid; k++,w=w*Wn) { //枚举左半部分
                complex x=A[j+k],y=w*A[j+mid+k];//蝴蝶效应
                A[j+k]=x+y;
                A[j+mid+k]=x-y;
            }
        }
    }
}
string sa,sb;
long long c[MAXN];
int n,m;
int main() {
```

```
    limit=1;
    l=0;
    cin>>n>>m;
    for(int i=0; i<=n; i++) cin>>a[i].x;
    for(int i=0; i<=m; i++) cin>>b[i].x;
    while(limit<=n+m) limit<<=1,l++;
    for(int i=0; i<limit; i++) r[i]= ( r[i>>1]>>1 )| ( (i&1)<<(l-1) ) ;
    fast_fast_tle(a,1);
    fast_fast_tle(b,1);
    for(int i=0; i<=limit; i++) a[i]=a[i]*b[i];
    fast_fast_tle(a,-1);
    for(int i=0; i<=n+m; i++) c[i]=(long long)(a[i].x/limit+0.5);
    for(int i=0; i<=n+m; i++) printf("%lld ",c[i]);
    return 0;
}
```