NTT（有模数下代替FFT

利用$\omega_n^1 = g^{\frac{n-1}{p}}$,用原根替代

```
#include<bits/stdc++.h>
inline int read()//必要时可以加
{
  register char ch=0;
  while(ch<48||ch>57)ch=getchar();
  return ch-'0';
}
using namespace std;
typedef long long ll;
const ll mod=998244353;//模数
const int maxn=2e6+5;
const ll G=3;   //原根
ll mypow(ll a,ll b){
    ll ans=1;
    while(b){
        if(b&1)ans=ans*a%mod;
        a=a*a%mod;b>>=1;
    }return ans;
}
const ll invG=mypow(G,mod-2);
int n,m,tr[maxn<<1];
ll f[maxn<<1],g[maxn<<1],invn;
void NTT(ll *f,bool op){
    for(int i=0;i<n;++i)
        if(i<tr[i])swap(f[i],f[tr[i]]);
    for(int p=2;p<=n;p<<=1){
        int len=p>>1;
        int tG=mypow(op?G:invG,(mod-1)/p);//$g^(p-1)/n$
        for(int k=0;k<n;k+=p){
            ll buf=1;
            for(int l=k;l<k+len;l++){
                int tt=buf*f[len+l]%mod;
                f[len+l]=f[l]-tt;
                if(f[len+l]<0)f[len+l]+=mod;
                f[l]=f[l]+tt;
                if(f[l]>mod)f[l]-=mod;
                buf=buf*tG%mod;
            }
        }
```

```
    }
}
int main(){
    scanf("%d%d",&n,&m);
    for(int i=0;i<=n;++i)scanf("%lld",&f[i].x);
    for(int i=0;i<=m;++i)scanf("%lld",&g[i].x);
    for(m+=n,n=1;n<=m;n<<=1);
    for(int i=0;i<n;++i)
        tr[i]=tr[i>>1]>>1|((i&1)?n>>1:0);
    NTT(f,1);NTT(g,1);
    for(int i=0;i<n;++i)f[i]=f[i]*g[i]%mod;
    NTT(f,0);
    invn=mypow(n,mod-2);
    for(int i=0;i<=m;++i)
        printf("%d ",(int)(f[i]*invn%mod));
}
```