拉格朗日插值

- 应用背景
  给出平面上n+1个点，求一条穿过这n+1个点的n次多项式，或这个多项式在另一个点处的值。
  显然可以高斯消元求出每一项系数，然后输出/直接爆算。
  拉格朗日插值分成两种：朴素的，和重心拉格朗日插值。
  一般情况下，朴素的和高斯消元在求解第1问时复杂度没有区别，
  但是后者无论第几问都可以用O(n^2)的复杂度爆艹高斯消元O(n^3)
- 板子

```
namespace polysum {    //orz dls
   const int D=101000;
   ll a[D],f[D],g[D],p[D],p1[D],p2[D],b[D],h[D][2],C[D];
   ll calcn(int d,ll *a,ll n) {//d次多项式(a[0-d])求第n项
       if (n<=d) return a[n];
       p1[0]=p2[0]=1;
       rep(i,0,d+1) {
           ll t=(n-i+mod)%mod;
           p1[i+1]=p1[i]*t%mod;
       }
       rep(i,0,d+1) {
           ll t=(n-d+i+mod)%mod;
           p2[i+1]=p2[i]*t%mod;
       }
       ll ans=0;
       rep(i,0,d+1) {
           ll t=g[i]*g[d-i]%mod*p1[i]%mod*p2[d-i]%mod*a[i]%mod;
           if ((d-i)&1) ans=(ans-t+mod)%mod;
           else ans=(ans+t)%mod;
       }
       return ans;
   }
   void init(int M) {//初始化预处理阶乘和逆元(取模乘法)
       f[0]=f[1]=g[0]=g[1]=1;
       rep(i,2,M+5) f[i]=f[i-1]*i%mod;
       g[M+4]=powmod(f[M+4],mod-2);
       per(i,1,M+4) g[i]=g[i+1]*(i+1)%mod;
   }
   ll polysum(ll n,ll *a,ll m) { // a[0].. a[m] \sum_{i=0}^{n-1} a[i]
                                 // m次多项式求第n项前缀和
       a[m+1]=calcn(m,a,m+1);
       rep(i,1,m+2) a[i]=(a[i-1]+a[i])%mod;
       return calcn(m+1,a,n-1);
   }
   ll qpolysum(ll R,ll n,ll *a,ll m) { // a[0].. a[m] \sum_{i=0}^{n-1} a[i]*R^i
       if (R==1) return polysum(n,a,m);
       a[m+1]=calcn(m,a,m+1);
       ll r=powmod(R,mod-2),p3=0,p4=0,c,ans;
       h[0][0]=0;h[0][1]=1;
       rep(i,1,m+2) {
           h[i][0]=(h[i-1][0]+a[i-1])*r%mod;
           h[i][1]=h[i-1][1]*r%mod;
       }
```

```
        rep(i,0,m+2) {
            ll t=g[i]*g[m+1-i]%mod;
            if (i&1) p3=((p3-h[i][0]*t)%mod+mod)%mod,p4=((p4-h[i][1]*t)%mod+mod)%mod;
            else p3=(p3+h[i][0]*t)%mod,p4=(p4+h[i][1]*t)%mod;
        }
        c=powmod(p4,mod-2)*(mod-p3)%mod;
        rep(i,0,m+2) h[i][0]=(h[i][0]+h[i][1]*c)%mod;
        rep(i,0,m+2) C[i]=h[i][0];
        ans=(calcn(m,C,n)*powmod(R,n)-c)%mod;
        if (ans<0) ans+=mod;
        return ans;
    }
}
...
```