HDU 4656 Evaluation（MTT）

题意

$x_k = bc^{2k} + d$

$F(x) = \sum_{i=0}^{n-1} a_i x^i$

给定 $\{a\}, b, c, d, n$ ，求 $F(x_0), F(x_1), \cdots, F(x_{n-1})$

思路

设 $ans_k = F(x_k)$

$$ans_k = \sum_{i=0}^{n-1} a_i (b \cdot c^{2k} + d)^i$$

二项展开得

$$ans_k = \sum_{i=0}^{n-1} \sum_{j=0}^{i} a_i \binom{i}{j} (b \cdot c^{2k})^j d^{i-j}$$

设 $f(i) = a_i d^i i!, g(j) = \dfrac{b^j}{d^j j!}$

$$ans_k = \sum_{i=0}^{n-1} \sum_{j=0}^{i} f(i) g(j) c^{2kj} \frac{1}{(i-j)!}$$

$$ans_k = \sum_{j=0}^{n-1} g(j) c^{2kj} \sum_{i=j}^{n-1} f(i) \frac{1}{(i-j)!}$$

不难发现，后面那个 $\sum$ 与 k 无关，可以预处理出来。

我们设 $h(j) = \sum_{i=j}^{n-1} f(i) \dfrac{1}{(i-j)!}$

这个可以直接用多项式乘出来。

有能力 $O(1)$ 求解 $h(j)$ 后，得到

$$ans_k = \sum_{j=0}^{n-1} g(j) h(j) c^{2jk}$$

$$ans_k = \sum_{j=0}^{n-1} g(j) h(j) c^{j^2 + k^2 - (k-j)^2}$$

$$ans_k = \sum_{j=0}^{n-1} g(j) h(j) \frac{c^{j^2} c^{k^2}}{c^{(k-j)^2}}$$

$$ans_k = c^{k^2} \sum_{j=0}^{n-1} g(j) h(j) c^{j^2} \frac{1}{c^{(k-j)^2}}$$

再进行一次多项式乘法就可以了。

代码

```
#include<bits/stdc++.h>
#define FOR(i,x,y) for(int i=(x),i##END=(y);i<=i##END;++i)
#define DOR(i,x,y) for(int i=(x),i##END=(y);i>=i##END;--i)
typedef long long ll;
using namespace std;
const double PI=acos(-1.0);
const int N=1<<18|5;
const int P=1e6+3;
namespace _Maths
```

```cpp
{
    ll gcd(ll a,ll b){return b?gcd(b,a%b):a;}
    void exgcd(ll a,ll b,ll &x,ll &y)
    {
        if(!b){x=1,y=0;return;}
        exgcd(b,a%b,y,x);y-=a/b*x;
    }
    ll Pow(ll a,ll p,ll P)
    {
        ll res=1;
        for(;p>0;(a*=a)%=P,p>>=1)if(p&1)(res*=a)%=P;
        return res;
    }
    ll inv(ll a,ll P){ll x,y;exgcd(a,P,x,y);return (x%P+P)%P;}
};
using namespace _Maths;
struct Complex
{
    double x,y;
    Complex operator +(const Complex &_)const{return (Complex){x+_.x,y+_.y};}
    Complex operator -(const Complex &_)const{return (Complex){x-_.x,y-_.y};}
    Complex operator *(const Complex &_)const{return (Complex){x*_.x-y*_.y,x*_.y+y*_.x};}
    Complex operator /(const int &_)const{return (Complex){x/_,y/_};}
};
namespace _Polynomial
{
    const int K=(1<<15)-1,L=15;
    Complex A[N<<1],B[N<<1],C[N<<1],D[N<<1];
    Complex w[N<<1];int r[N<<1];
    void DFT(Complex *a,int op,int n)
    {
        FOR(i,0,n-1)if(i<r[i])swap(a[i],a[r[i]]);
        for(int i=2;i<=n;i<<=1)
            for(int j=0;j<n;j+=i)
                for(int k=0;k<i/2;k++)
                {
                    Complex u=a[j+k],t=w[(op==1?n/i*k:n-n/i*k)&(n-1)]*a[j+k+i/2];
                    a[j+k]=u+t,a[j+k+i/2]=u-t;
                }
        if(op==-1)FOR(i,0,n-1)a[i]=a[i]/n;
    }
    void multiply(const int *a,const int *b,int *c,int n1,int n2)
    {
        int n=1;
        while(n<n1+n2-1)n<<=1;
        FOR(i,0,n1-1)A[i].x=a[i]&K,A[i].y=a[i]>>L;
        FOR(i,0,n2-1)B[i].x=b[i]&K,B[i].y=b[i]>>L;
        FOR(i,n1,n-1)A[i].x=A[i].y=0;
        FOR(i,n2,n-1)B[i].x=B[i].y=0;
        FOR(i,0,n-1)r[i]=(r[i>>1]>>1)|((i&1)*(n>>1));
        FOR(i,0,n-1)w[i]=(Complex){cos(2*PI*i/n),sin(2*PI*i/n)};

        DFT(A,1,n),DFT(B,1,n);
        FOR(i,0,n-1)
        {
            int j=(n-i)&(n-1);
            C[i]=(Complex){0.5*(A[i].x+A[j].x),0.5*(A[i].y-A[j].y)}*B[i];
            D[i]=(Complex){0.5*(A[i].y+A[j].y),0.5*(A[j].x-A[i].x)}*B[i];
        }
        DFT(C,-1,n),DFT(D,-1,n);
        FOR(i,0,n-1)
        {
            ll s=C[i].x+0.5,t=C[i].y+0.5,u=D[i].x+0.5,v=D[i].y+0.5;
            c[i]=(s%P+((t+u)%P<<L)+(v%P<<L<<L))%P;
        }
    }
};
int fac[N],ifac[N],h[N];
int A[N],B[N],C[N<<1];
```

```
int a[N],b,c,d;
int n;

ll f(int i){return (ll)a[i]*Pow(d,i,P)%P*fac[i]%P;}
ll g(int i){return (ll)Pow(b,i,P)*inv(Pow(d,i,P)%P*fac[i]%P,P)%P;}

int main()
{
    fac[0]=fac[1]=1;FOR(i,2,N-1)fac[i]=(ll)fac[i-1]*i%P;
    ifac[0]=ifac[1]=1;FOR(i,2,N-1)ifac[i]=(ll)(P-P/i)*ifac[P%i]%P;
    FOR(i,2,N-1)ifac[i]=(ll)ifac[i-1]*ifac[i]%P;
    scanf("%d%d%d%d",&n,&b,&c,&d);
    FOR(i,0,n-1)scanf("%d",&a[i]);

    FOR(i,0,n-1)A[i]=f(i);
    FOR(i,1-n,0)B[(n-1)+i]=ifac[-i];
    _Polynomial::multiply(A,B,C,n,n);
    FOR(i,0,n-1)h[i]=C[(n-1)+i];

    FOR(i,0,n-1)A[i]=g(i)*h[i]%P*Pow(c,(ll)i*i,P)%P;
    FOR(i,1-n,n-1)B[(n-1)+i]=inv(Pow(c,(ll)i*i,P),P);
    _Polynomial::multiply(A,B,C,n,2*n-1);
    FOR(i,0,n-1)printf("%lld\n",Pow(c,(ll)i*i,P)*C[(n-1)+i]%P);
    return 0;
}
```


作者： Paulliant
出处：https://www.cnblogs.com/Paulliant/p/10272296.html
版权：本作品采用「署名-非商业性使用-相同方式共享 4.0 国际」许可协议进行许可。

分类: 出处--HDU ， 算法--多项式

« 上一篇： HDU 5829 Rikka with Subset（NTT）
» 下一篇： HDU 5552 Bus Routes（NTT+分治）