

数论 5

卢卡斯定理

算法简介

$O(p + \log_p n)$ 计算 $\binom{n}{m} \bmod p$ 的算法， p 为素数。

算法实现

$$\binom{p}{i} = \frac{p!}{i!(p-i)!}$$

于是对 $1 \leq i \leq p-1$ 有 $\binom{p}{i} \equiv 0 \pmod p$

$$(1+x)^p \equiv \binom{p}{0} + \binom{p}{1}x + \dots + \binom{p}{p}x^p \equiv 1+x^p \pmod p$$

对 $\binom{n}{m}$ 不妨设 $n=k_1p+r_1, m=k_2p+r_2$ 于是有

$$(1+x)^n \equiv (1+x)^{k_1p+r_1} \equiv ((1+x)^p)^{k_1} (1+x)^{r_1} \equiv (1+x^p)^{k_1} (1+x)^{r_1} \pmod p$$

对比左右两边 x_m 的系数，有

$$\binom{n}{m} \equiv \binom{k_1}{k_2} \binom{r_1}{r_2} \pmod p$$

于是递归处理，有

$$\text{Lucas}(n, m) \equiv \text{Lucas}\left(\left\lfloor \frac{n}{p} \right\rfloor, \left\lfloor \frac{m}{p} \right\rfloor\right) \binom{n \bmod p}{m \bmod p}$$

代码模板

[洛谷p3807](#)

```
const int MAXN=1e5+5;
int frac[MAXN],invfrac[MAXN];
int quick_pow(int a,int b,int p){
    int ans=1;
    while(b){
        if(b&1)ans=1LL*ans*a%p;
        a=1LL*a*a%p;
        b>>=1;
    }
    return ans;
}
int Lucas(int n,int m,int p){
```

```

    if(n<m)return 0;
    else if(n<p)return 1LL*frac[n]*invfrac[n-m]%p*invfrac[m]%p;
    else return 1LL*Lucas(n%p,m%p,p)*Lucas(n/p,m/p,p)%p;
}
int main()
{
    int T=read_int();
    while(T--){
        int n=read_int(),m=read_int(),p=read_int();
        frac[0]=1;
        _for(i,1,p)frac[i]=1LL*frac[i-1]*i%p;
        invfrac[p-1]=quick_pow(frac[p-1],p-2,p);
        for(int i=p-1;i;i--)
            invfrac[i-1]=1LL*invfrac[i]*i%p;
        enter(Lucas(n+m,n,p));
    }
    return 0;
}

```

拓展卢卡斯

算法简介

$O\left(\sum p_i^k + \sum \log_2 n \log_{p_i^k} n\right)$ 计算 $\binom{n}{m} \bmod p$ 的算法，其中 $p = \prod p_i^k$

算法实现

考虑计算 $\binom{n}{m} \equiv x_i \pmod{p_i^k}$ 然后中国剩余定理合并答案。

$$\binom{n}{m} \equiv \frac{n!}{m!(n-m)!} \pmod{p^k}$$

分子不一定与模数互质，不能直接使用逆元计算。考虑提取出 $n!$ 所有的 p 因子，记为 $g(n)$ 同时记 $f(n) = \frac{n!}{p^{g(n)}}$ 代入上式，有

$$\frac{\frac{n!}{p^{g(n)}}}{\frac{m!}{p^{g(m)}} \frac{(n-m)!}{p^{g(n-m)}}} p^{g(n)-g(m)-g(n-m)} \equiv \frac{f(n)}{f(m)f(n-m)} p^{g(n)-g(m)-g(n-m)} \pmod{p^k}$$

于是有 $(f(n), p^k) = 1$ 可以参与逆元计算。设 $n = ap^k + b$

$$n! = \prod_{i=1, p \nmid i}^{p \mid i} n_i \prod_{i=1, p \nmid i}^{p \nmid i} n_i = \left(\left\lfloor \frac{np}{p} \right\rfloor!\right) p^{\left\lfloor \frac{np}{p} \right\rfloor} \left(\prod_{i=1, p \nmid i}^{p \nmid i} p^k\right)^a \prod_{i=1, p \nmid i}^{p \nmid i} b_i$$

于是有

$$f(n) = f\left(\left\lfloor \frac{np}{p} \right\rfloor\right) \left(\prod_{i=1, p \nmid i}^{p \nmid i} p^k\right)^a \prod_{i=1, p \nmid i}^{p \nmid i} b_i$$

$$g(n) = g\left(\left\lfloor \frac{np}{p} \right\rfloor\right) + \left\lfloor \frac{np}{p} \right\rfloor$$

经过 $\sum p_i^k$ 预处理后即可 $\sum \log_2 n \log_{p^k} n$ 递归计算答案。

代码模板

洛谷p4720

```
int quick_pow(int a,int b,int p){
    int ans=1;
    while(b){
        if(b&1)ans=1LL*ans*a%p;
        a=1LL*a*a%p;
        b>>=1;
    }
    return ans;
}

void exgcd(LL a,LL b,LL &tx,LL &ty){
    if(b==0){
        tx=1,ty=0;
        return;
    }
    exgcd(b,a%b,ty,tx);
    ty-=a/b*tx;
}

int inv(int x,int mod){
    LL t1,t2;
    exgcd(x,mod,t1,t2);
    return t1%mod;
}

namespace Lucas{
    const int MAXM=30;
    vector<int> frac[MAXM];
    int p_cnt,prime[MAXM],y[MAXM],pod[MAXM];
    void Init(int p,int mod){
        frac[p_cnt].resize(mod+1);
        frac[p_cnt][0]=1;
        _rep(i,1,mod){
            if(i%p)frac[p_cnt][i]=1LL*frac[p_cnt][i-1]*i%mod;
            else frac[p_cnt][i]=frac[p_cnt][i-1];
        }
        prime[p_cnt]=p;
        pod[p_cnt]=mod/p*(p-1);
        y[p_cnt++]=mod;
    }
    void init(int mod){
        p_cnt=0;
        int t=mod;
        for(int p=2;p*p<=t;p++){
            if(t%p==0){
                int mod2=1;
```

```
        while(t%p==0)t/=p,mod2*=p;
        Init(p,mod2);
    }
}
if(t!=1)Init(t,t);
}
int f(LL n,int i){
    int ans=1;
    while(n){
ans=1LL*ans*quick_pow(frac[i][y[i]],(n/y[i])%pod[i],y[i])%y[i]*frac[i][n%y[
i]]%y[i];
        n/=prime[i];
    }
    return ans;
}
int g(LL n,int i){
    int ans=0;
    while(n)ans+=(n/=prime[i]);
    return ans;
}
int cal(LL n,LL m,int i){
    int t=1LL*f(n,i)*inv(1LL*f(m,i)*f(n-m,i)%y[i],y[i])%y[i];
    t=1LL*t*quick_pow(prime[i],g(n,i)-g(m,i)-g(n-m,i),y[i])%y[i];
    return t;
}
int C(LL n,LL m){
    int mod=1,ans=0;
    _for(i,0,p_cnt)mod*=y[i];
    _for(i,0,p_cnt){
        int x=cal(n,m,i);
        ans=(ans+1LL*x*(mod/y[i])%mod*inv(mod/y[i],y[i]))%mod;
    }
    return (ans+mod)%mod;
}
}
int main()
{
    LL n=read_LL(),m=read_LL();
    int mod=read_int();
    Lucas::init(mod);
    enter(Lucas::C(n,m));
    return 0;
}
```

From:

<https://wiki.cvbbacm.com/> - **CVBB ACM Team**

Permanent link:

https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:%E6%95%B0%E8%AE%BA_5 

Last update: **2020/10/16 15:01**