拓展中国剩余定理

作用

解决模数不互质的问题

原理

合并和迭代思想

$$\text{比如我这里有一个同余方程}\begin{cases}x\equiv a_1 \mod m_1\\x\equiv a_2 \mod m_2\end{cases}$$

$$\text{那么}m_1x_1+a_1=m_2x_2+a_2$$
$$m_1x_1=(a_2-a_1)+m_2x_2$$
$$m_1x_1=a_2-a_1 \mod m_2$$

斐波那契数

递归方法定义$F(0)=0$，$F(1)=1,F(n)=F(n-1)+F(n-2)$，$n\geq 2$

用待定系数法推导通项公式：(推导过程繁琐，利用等比数列，可自行查阅资料)

$F(n)=\frac{\sqrt{5}}{5}[(\frac{1+\sqrt{5}}{2})^n-(\frac{1-\sqrt{5}}{2})^n]$

更普遍的形式：

$$\text{若}F(n)=\begin{cases}f_0,n=0\\f_1,n=1\\a*F(n-1)+b*F(n-2),n>1\end{cases}$$
$$\text{那么}F(n)=\frac{k^n(f_1-mf_0)-m^n(f_1-kf_0)}{k-m}$$
$$\text{其中}k,m=\frac{a\pm\sqrt{a^2+4b}}{2}$$

斐波那契第一列通项公式(参考AHOI2004数字迷阵，第二行第一列的元素为前i-1行未出现的最小正整数，而A[i,2]=2A[i,1]-(i-1)

$f[i]=trunc(i*t+i-1),\text{其中}t=(1+\sqrt{5})/2$

卡特兰数

Catalan Number，是组合数学中一个经常出现在技术问题的数列。

前几项为：1,2,5,14,42,132,429,1430,4862,16796,58786,208012,742900...

求卡特兰数列第n项

$$1.\text{递归公式1} \quad f(n)=\sum_{i=0}^{n-1} f(i)\times f(n-i-1)$$
$$2.\text{递归公式2} \quad f(n)=\frac{f(n-1)*(4*n-2)}{n+1}$$
$$3.\text{组合公式1} \quad f(n)=\frac{C_{2n}^{n}}{n+1}$$
$$4.\text{组合公式2} \quad f(n)=C_{2n}^{n}-C_{2n}^{n-1}$$

常见的案例

左右括号、二叉树计算、欧拉多边形分割

斯特林数

第一类斯特林数

表示将n个不同元素构成m个圆排列的数目。

$S(n,m)=S(n-1,m-1)+nS(n-1,m)$

第二类斯特林数

表示将n个不同元素拆分成m个集合的方案数

$S(n,m)=S(n-1,m-1)+mS(n-1,m)$

卢卡斯定理

可以用来求$C(n,m)modp$的值，其中n和m是非负整数，p是素数。

一般用于m,n很大而p很小，或者n,m不大但大于p，这样用阶乘就解决不了。

结论1

$$Lucas(n,m,p)=C(n\%p,m\%p)*Lucas(n/p,m/p,p),Lucas(x,0,p)=1;$$
$$\text{计算组合数该用逆元：}C(a,b)=(a!/(a-b)!)*(b!)^{(p-2)} \mod p$$

结论2

把n写成p进制a[n]a[n-1]a[n-2]...a[0]，把m写成p进制b[n]b[n-1]b[n-2]...b[0]，则C(n,m)与C(a[n],b[n])*C(a[n-1],b[n-1])*C(a[n-2],b[n-2])*...*C(a[0],b[0])模p同余

```
#include<bits/stdc++.h>
#define int long long
using namespace std;

int t,n,m,p;

int fpow(int a,int b){
    int res=1;
    while(b){
        if(b&1) res=res*a%p;
        a=a*a%p;
```

```
        b>>=1;
    }
    return res;
}

int C(int a,int b){
    if(a<b) return 0;
    if(b>a-b) b=a-b;
    int s1=1,s2=1;
    for(int i=0;i<b;i++){
        s1=s1*(a-i)%p; //a!/(a-b)!
        s2=s2*(i+1)%p; //b!
    }
    return s1*fpow(s2,p-2)%p; //逆元
}

int Lucas(int a,int b){
    if(b==0) return 1;
    return C(a%p,b%p)*Lucas(a/p,b/p)%p;
}

signed main(){
    ios::sync_with_stdio(0);
    cin.tie(0);cout.tie(0);
    cin>>t;
    while(t--){
        cin>>n>>m>>p;
        cout<<Lucas(n+m,n)%p<<endl;
    }
    return 0;
}
```

扩展卢卡斯定理

在不保证p是质数的情况下，求$C_n^m \mod p$

luogu[P4720 【模板】扩展卢卡斯定理/exLucas]

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
template<typename T> inline bool chkmin(T &a, T b) { return b < a ? a = b, 1 :
0; }
template<typename T> inline bool chkmax(T &a, T b) { return b > a ? a = b, 1 :
0; }

ll n, m, p;

void Exgcd(ll a, ll b, ll &x, ll &y){ //扩展欧几里得
    if (!b) x=1,y=0;
    else Exgcd(b,a%b,y,x),y-=a/b*x;
}

inline ll fpm(ll x, ll power, ll Mod){ //快速幂
    ll res = 1;
```

```
    for (; power; power>>= 1,(x*=x)%=Mod)
        if(power&1) (res*=x)%=Mod;
    return res;
}

inline ll fac(ll n, ll pi, ll pk){ // 求阶乘
    if(!n) return 1;
    ll res=1;
    for(int i=2;i<=pk;i++) if(i%pi) (res*=i)%=pk;
    res=fpm(res,n/pk,pk);
    for(int i=2;i<=n%pk;i++) if(i%pi) (res*=i)%=pk;
    return res*fac(n/pi,pi,pk)%pk;
}

inline ll Inv(ll n, ll Mod){ //求逆元
    ll x, y; Exgcd(n, Mod, x, y);
    return (x % Mod + Mod) % Mod;
}

inline ll CRT(ll b, ll Mod){ //中国剩余定理合并答案
    return b*Inv(p/Mod, Mod)%p*(p/Mod)%p;
}

inline ll factor(ll x, ll Mod) { //求x!中包含Mod因子的数量
    return x?factor(x/Mod,Mod)+(x/Mod):0;
}

inline ll Comb(ll n,ll m,ll pi,ll pk){ //求组合数,pi为模数p的因子
    ll k=factor(n,pi)-factor(m,pi)-factor(n-m,pi);
    if (!fpm(pi,k,pk)) return 0;
    return fac(n,pi,pk)*Inv(fac(m,pi,pk),pk)%pk*Inv(fac(n-
m,pi,pk),pk)%pk*fpm(pi,k,pk)%pk;
}

inline ll ExLucas(ll n, ll m){ //扩展卢卡斯定理
    ll res=0,tmp=p;
    for(int i=2;i<=sqrt(p+.5);i++) if(!(tmp%i)){ //枚举模数因子
        ll pk=1;
        while(!(tmp%i)) pk*=i,tmp/=i;
        (res+=CRT(Comb(n,m,i,pk),pk))%=p; //合并所有结果
    }
    if(tmp>1) (res+=CRT(Comb(n,m,tmp,tmp),tmp))%=p; //不要漏掉
    return res;
}

signed main(){
    cin>>n>>m>>p;
    printf("%lld\n",ExLucas(n,m));
    return 0;
}
```

参考资料

https://www.bilibili.com/video/BV1764y1R7kj

https://www.bilibili.com/video/BV1xb411x74F?

https://www.cnblogs.com/zjp-shadow/p/9267675.html#autoid-3-3-0