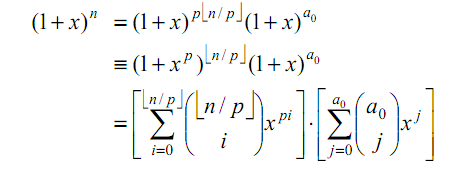
（1）Lucas定理：p为素数，则有:

http://pic002.cnblogs.com/images/2012/361759/2012112411530924.jpg

http://pic002.cnblogs.com/images/2012/361759/2012112411532478.jpg

http://pic002.cnblogs.com/images/2012/361759/2012112411535916.jpg

（2）证明： n=(ak...a2,a1,a0)p = (ak...a2,a1)p\*p + a0 =  [n/p]\*p+a0，m=[m/p]\*p+b0其次，我们知道，对任意质数p有(1+x)^p=1+(x^p)(mod p) 。我们只要证明这个式子：C(n,m)=C([n/p],[m/p]) \* C(a0,b0)(mod p)，那么就可以用归纳法证明整个定理。对于模p而言，我们有下面的式子成立：



上式左右两边的x的某项x^m(m<=n)的系数对模p同余。其中左边的x^m的系数是 C(n,m)。 而由于a0和b0都小于p，因此右边的x^m 一定是由 x^([m/p]\*p) 和 x^b0 (即i=[m/p] , j=b0 ) 相乘而得 因此有：C(n,m)=C([n/p],[m/p]) \* C(a0,b0)  (mod p)。

（3）拓展应用：上面的p是素数，那么不是素数怎么办呢？若不是素数，将p分解质因数，将C(n,m)分别按照（1）中的方法求对p的质因数的模，然后用中国剩余定理合并。比如计算C(10,3)%14。C(10,3)=120,14有两个质因数2和7，120%2=0,120%7=1,这样用(2,0)(7,1)找到最小的正整数8即是答案，即C(10,3)%14=8。注意，这里只适用于p分解完质因数后每个质因数只出现一次，例如12=2\*2\*3就不行，因为2出现了两次。若p分解完质因数后，含有某个质因数出现多次，比如C(10,3)%98,其中98=2\*7\*7,此时就要把7\*7看做一个数,即:120%2=0,120%49=22,用(2,0)(49,22)和中国剩余定理得到答案22，即C(10,3)%98=22。此时，你又会有疑问，C(10,3)%49不也是模一个非素数吗？此时不同的是这个非素数不是一般的非素数，而是某个素数的某次方。下面（4）介绍如何计算C(n,m)%p^t(t>=2,p为素数)。

（4）计算C(n,m)%p^t。我们知道，C(n,m)=n!/m!/(n-m)!，若我们可以计算出n!%p^t，我们就能计算出m!%p^t以及(n-m)!%p^t。我们不妨设x=n!%p^t,y=m!%p^t,z=(n-m)!%p^t,那么答案就是x\*reverse(y,p^t)\*reverse(z,p^t)(reverse(a,b)计算a对b的乘法逆元)。那么下面问题就转化成如何计算n!%p^t。比如p=3,t=2,n=19,

n!=1\*2\*3\*4\*5\*6\*7\*8\* ……\*19

   =[1\*2\*4\*5\*7\*8\*… \*16\*17\*19]\*(3\*6\*9\*12\*15\*18)

   =[1\*2\*4\*5\*7\*8\*… \*16\*17\*19]\*3^6(1\*2\*3\*4\*5\*6)

然后发现后面的是(n/p)!，于是递归即可。前半部分是以p^t为周期的[1\*2\*4\*5\*7\*8]=[10\*11\*13\*14\*16\*17](mod 9)。下面是孤立的19，可以知道孤立出来的长度不超过 p^t,于是暴力即可。那么最后剩下的3^6啊这些数怎么办呢？我们只要计算出n!,m!,(n-m)!里含有多少个p(不妨设a,b,c)，那么a-b-c就是C(n,m)中p的个数，直接算一下就行。

至此整个计算C(n,m)%p(p为任意数)的问题完美解决。下面给出代码：

i64 POW(i64 a,i64 b,i64 mod)

{

i64 ans=1;

while(b)

{

if(b&1) ans=ans\*a%mod;

a=a\*a%mod;

b>>=1;

}

return ans;

}

i64 POW(i64 a,i64 b)

{

i64 ans=1;

while(b)

{

if(b&1) ans=ans\*a;

a=a\*a;

b>>=1;

}

return ans;

}

i64 exGcd(i64 a,i64 b,i64 &x,i64 &y)

{

i64 t,d;

if(!b)

{

x=1;

y=0;

return a;

}

d=exGcd(b,a%b,x,y);

t=x;

x=y;

y=t-a/b\*y;

return d;

}

bool modular(i64 a[],i64 m[],i64 k)

{

i64 d,t,c,x,y,i;

for(i=2;i<=k;i++)

{

d=exGcd(m[1],m[i],x,y);

c=a[i]-a[1];

if(c%d) return false;

t=m[i]/d;

x=(c/d\*x%t+t)%t;

a[1]=m[1]\*x+a[1];

m[1]=m[1]\*m[i]/d;

}

return true;

}

i64 reverse(i64 a,i64 b)

{

i64 x,y;

exGcd(a,b,x,y);

return (x%b+b)%b;

}

i64 C(i64 n,i64 m,i64 mod)

{

if(m>n) return 0;

i64 ans=1,i,a,b;

for(i=1;i<=m;i++)

{

a=(n+1-i)%mod;

b=reverse(i%mod,mod);

ans=ans\*a%mod\*b%mod;

}

return ans;

}

i64 C1(i64 n,i64 m,i64 mod)

{

if(m==0) return 1;

return C(n%mod,m%mod,mod)\*C1(n/mod,m/mod,mod)%mod;

}

i64 cal(i64 n,i64 p,i64 t)

{

if(!n) return 1;

i64 x=POW(p,t),i,y=n/x,temp=1;

for(i=1;i<=x;i++) if(i%p) temp=temp\*i%x;

i64 ans=POW(temp,y,x);

for(i=y\*x+1;i<=n;i++) if(i%p) ans=ans\*i%x;

return ans\*cal(n/p,p,t)%x;

}

i64 C2(i64 n,i64 m,i64 p,i64 t)

{

i64 x=POW(p,t);

i64 a,b,c,ap=0,bp=0,cp=0,temp;

for(temp=n;temp;temp/=p) ap+=temp/p;

for(temp=m;temp;temp/=p) bp+=temp/p;

for(temp=n-m;temp;temp/=p) cp+=temp/p;

ap=ap-bp-cp;

i64 ans=POW(p,ap,x);

a=cal(n,p,t);

b=cal(m,p,t);

c=cal(n-m,p,t);

ans=ans\*a%x\*reverse(b,x)%x\*reverse(c,x)%x;

return ans;

}

//计算C(n,m)%mod

i64 Lucas(i64 n,i64 m,i64 mod)

{

i64 i,t,cnt=0;

i64 A[205],M[205];

for(i=2;i\*i<=mod;i++) if(mod%i==0)

{

t=0;

while(mod%i==0)

{

t++;

mod/=i;

}

M[++cnt]=POW(i,t);

if(t==1) A[cnt]=C1(n,m,i);

else A[cnt]=C2(n,m,i,t);

}

if(mod>1)

{

M[++cnt]=mod;

A[cnt]=C1(n,m,mod);

}

modular(A,M,cnt);

return A[1];

}