

# 卢卡斯定理

## Lucas Theorem

### 卢卡斯定理

Theorem: 设  $p$  为素数, 则

$$\binom{n}{m} \bmod p = \binom{\lfloor n/p \rfloor}{\lfloor m/p \rfloor} \times \binom{n \bmod p}{m \bmod p} \bmod p$$

Understanding: 将  $n, m$  均写作  $p$  进制数, 则  $\binom{n}{m} \bmod p$  等于取出  $n, m$  的每一位做组合数后相乘。

Correctness: 暂略。

Application: 组合数取模: 求解当  $p$  为素数且  $n, m$  较大时的  $\binom{n}{m} \bmod p$ 。

Code:

```
1 LL C(LL n, LL m, LL p){
2     if(m > n) return 0;
3     return fac[n] * inv[fac[m]] % p * inv[fac[n-m]] % p;
4 }
5
6 LL lucas(LL n, LL m, LL p){
7     if(m == 0) return 1;
8     return C(n%p, m%p, p) * lucas(n/p, m/p, p) % p;
9 }
```

### 扩展卢卡斯定理

设  $p$  是正整数 (不保证是素数), 则  $\binom{n}{m} \bmod p$  的求解步骤如下:

设  $p = \prod_{i=1}^r p_i^{k_i}$ , 则对于每个  $p_i^{k_i}$ , 若可以求出每一个  $\binom{n}{m} \bmod p_i^{k_i}$ , 那么对于同余方程组:

$$\begin{cases} \binom{n}{m} \equiv a_1 \pmod{p_1^{k_1}} \\ \binom{n}{m} \equiv a_2 \pmod{p_2^{k_2}} \\ \dots \\ \binom{n}{m} \equiv a_r \pmod{p_r^{k_r}} \end{cases}$$

使用中国剩余定理合并答案即可。所以问题转化为求解  $\binom{n}{m} \bmod p^k$ , 即  $\frac{n!}{m!(n-m)!} \bmod p^k$ , 其中  $p$  为质数。

可惜  $p^k$  不一定是质数,  $m!, (n-m)!$  不一定存在逆元, 不可直接求解。设  $n!$  中有  $x$  个  $p$  因子,  $m!$  中有  $y$  个  $p$  因子,  $(n-m)!$  中有  $z$  个  $p$  因子, 则

$$\frac{n!}{m!(n-m)!} \bmod p^k = \frac{\frac{n!}{p^x}}{\frac{m!}{p^y} \frac{(n-m)!}{p^z}} p^{x-y-z} \bmod p^k$$

现在  $\frac{m!}{p^y}, \frac{(n-m)!}{p^z} \perp p^k$ , 可以用逆元解了。所以问题转化成求:  $\frac{n!}{p^x} \bmod p^k$ 。

注意到:

$$\begin{aligned}
n! &\equiv 1 \cdot 2 \cdots n \\
&\equiv \left( p \cdot 2p \cdots \left\lfloor \frac{n}{p} \right\rfloor p \right) (1 \cdot 2 \cdots) \\
&\equiv p^{\left\lfloor \frac{n}{p} \right\rfloor} \left( \left\lfloor \frac{n}{p} \right\rfloor \right)! \cdot \prod_{i=1}^n [p \nmid i] i \\
&\equiv p^{\left\lfloor \frac{n}{p} \right\rfloor} \left( \left\lfloor \frac{n}{p} \right\rfloor \right)! \cdot \left( \prod_{i=1}^{p^k} [p \nmid i] i \right)^{\left\lfloor \frac{n}{p^k} \right\rfloor} \cdot \left( \prod_{i=p^k}^n [p \nmid i] i \right) \pmod{p^k} \\
&\equiv p^{\left\lfloor \frac{n}{p} \right\rfloor} \left( \left\lfloor \frac{n}{p} \right\rfloor \right)! \cdot \left( \prod_{i=1}^{p^k} [p \nmid i] i \right)^{\left\lfloor \frac{n}{p^k} \right\rfloor} \cdot \left( \prod_{i=1}^{n \bmod p^k} [p \nmid i] i \right) \pmod{p^k}
\end{aligned}$$

关于乘积式的变化：注意周期性，前一个乘积是周期，后一个是余项。例如，在  $\bmod 3^2$  下，

$$\begin{aligned}
22! &\equiv 1 \cdots 22 \\
&\equiv (3 \cdot 6 \cdots 21)(1 \cdot 2 \cdot 4 \cdot 5 \cdot 7 \cdot 8)(10 \cdot 11 \cdot 13 \cdot 14 \cdot 16 \cdot 17)(19 \cdot 20 \cdot 22) \\
&\equiv 3^7 \cdot 7! \cdot (1 \cdot 2 \cdot 4 \cdot 5 \cdot 7 \cdot 8)^2 (19 \cdot 20 \cdot 22) \pmod{3^2}
\end{aligned}$$

注意我们要求的是  $\frac{n!}{p^x} \bmod p^k$ ，即把  $n!$  中  $p$  的因子全除掉，而上式中  $p^{\left\lfloor \frac{n}{p} \right\rfloor}$  除掉后，还应把  $\left( \left\lfloor \frac{n}{p} \right\rfloor \right)!$  中  $p$  的因子除掉。不妨设  $f(n) = \frac{n!}{p^x}$ ，则上式写作：

$$f(n) \equiv f\left(\left\lfloor \frac{n}{p} \right\rfloor\right) \cdot \left( \prod_{i=1}^{p^k} [p \nmid i] i \right)^{\left\lfloor \frac{n}{p^k} \right\rfloor} \cdot \left( \prod_{i=1}^{n \bmod p^k} [p \nmid i] i \right) \pmod{p^k}$$

这是一个递归式，最多递归  $\log_p n$  层，后面两个乘积直接  $O(p^k)$  暴力计算。

我们还需要计算  $p^{x-y-z}$ ，也就是计算  $x, y, z$ ，设  $x = g(n)$  表示  $n!$  的因子  $p$  的个数，那么  $g(n) = g\left(\left\lfloor \frac{n}{p} \right\rfloor\right) + \left\lfloor \frac{n}{p} \right\rfloor$ ，解得： $g(n) = \sum_{i \geq 1} \left\lfloor \frac{n}{p^i} \right\rfloor$ （这个式子《具体数学》里也有推导）， $O(\log_p n)$  可算。

综上，我们最后要求的就是：

$$\binom{n}{m} \bmod p^k = \frac{f(n)}{f(m) \cdot f(n-m)} \cdot p^{g(n)-g(m)-g(n-m)} \bmod p^k$$

Code:

```

1  #include<bits/stdc++.h>
2
3  using namespace std;
4
5  typedef long long LL;
6
7  inline LL fpow(LL bs, LL idx, LL mod){
8      LL res = 1;
9      while(idx){
10         if(idx & 1) (res *= bs) %= mod;
11         idx >>= 1;
12         (bs *= bs) %= mod;
13     }
14     return res;
15 }
16
17 LL exgcd(LL a, LL b, LL &x, LL &y){
18     if(b == 0){ x = 1, y = 0; return a; }
19     LL d = exgcd(b, a % b, x, y);
20     LL t = x; x = y; y = t - a / b * y;
21     return d;
22 }
23
24 inline LL inv(LL x, LL mod){
25     LL res, y;
26     exgcd(x, mod, res, y);
27     ((res %= mod) += mod) %= mod;
28     return res;
29 }
30
31 LL F(LL n, LL p, LL pk){
32     if(n == 0) return 1;
33     LL res = 1;

```

```

34     for(LL i = 1; i <= pk; i++){
35         if(i % p == 0) continue;
36         (res *= i) %= pk;
37     }
38     res = fpow(res, n / pk, pk);
39     for(LL i = 1; i <= n % pk; i++){
40         if(i % p == 0) continue;
41         (res *= i) %= pk;
42     }
43     return res * F(n / p, p, pk) % pk;
44 }
45
46 inline LL calc(LL n, LL m, LL p, LL pk){
47     // calculate C(n,m) % pi^k
48     LL res = 0;
49     for(LL i = n; i; i /= p) res += i / p;
50     for(LL i = m; i; i /= p) res -= i / p;
51     for(LL i = n - m; i; i /= p) res -= i / p;
52     res = fpow(p, res, pk);
53     res = F(n, p, pk) * inv(F(m, p, pk), pk) % pk * inv(F(n-m, p, pk), pk) % pk * res % pk;
54     return res;
55 }
56
57 inline LL exLucas(LL n, LL m, LL p){
58     // calculate C(n,m) % p
59     LL P = p, res = 0;
60     for(LL i = 2; i * i <= p; i++){
61         if(p % i) continue;
62         LL pk = 1;
63         while(p % i == 0) p /= i, pk *= i;
64         (res += calc(n, m, i, pk) * (P / pk) % P * inv(P / pk, pk) % P) %= P;
65     }
66     if(p > 1)
67         (res += calc(n, m, p, p) * (P / p) % P * inv(P / p, p) % P) %= P;
68     return res;
69 }
70
71 int main(){
72     LL n, m, p; scanf("%lld%lld%lld", &n, &m, &p);
73     printf("%lld\n", exLucas(n, m, p));
74     return 0;
75 }

```