# 求组合数C(n,m)%P:

## 预处理阶乘：

先预处理阶乘, 再直接根据公式

n!/(m!(n - m)!) 要求p是int范围内素数，n,m小于1e6 且p>n>=m

涉及到求阶乘对P逆元问题，用扩展欧几里得或者费马小定理，求逆元

时间复杂度是：n的预处理+logn的查询。

用递推求逆元是：3\*n的预处理+o(1)的查询

## 卢卡斯定理：

**C(n,m)%p=C(n/p,m/p)\*C(n%p,m%p)%p**

这里要求p是素数， p小于1e6 但n和m很大

p>n>=m时，用预处理阶乘方式求解 ，否则利用这个分治算法，缩小n和m范围

## 扩展卢卡斯定理：

p不一定是素数，p小于1e6 但n和m很大

此时逆元可能不存在，也没法预处理阶乘了，怎么做呢。

设P=p1^c1 \* p2^c2 \* ….. \*pk^ck 代表P被分解成素数次方相乘形式，P有k个素数因子

假设我们已经求出了C(n,m)% p1^c1 C(n,m)% p2^c2 …… C(n,m)% pk^ck

可以用扩展中国剩余合并这些模方程组求得答案。

现在问题就转行成了：C(n,m)% p^c如何求得，这里的p是小于1e6的素数，且c很小

C(n,m)% p^c=n!/(m!(n - m)!)%p^c;现在问题变为,求出几个阶乘和阶乘的逆元

对于求n! %p^c,在区间[1,n]显然存在ceil(n/p)个数是p倍数，把每个p的倍数约去p,再乘起来，得到ceil(n/p)！

到这里，有的人说，求CC(n,m)%p^c,要是n>= p^c,不是直接得0么，刚才分解做什么。

实际上n>= p^c,时CC(n,m) %p^c不一定是0，因为求CC过程有除法，会约掉一部分p,

写到这，从n!提出p的倍数的数作用很明显了，就是和分母中m!与(n-m)!的约掉。

此时若分母中剩余的p数量大于等于c,那就得0；

否则意味着分母和p^c互质, 那问题变成了如何求n和m很大的情况下，约完后的n! %p^c

由于n可能很大，绝对不能暴力求,n!对p^c取模意味着，区间[1,n]每个数都要对p^c取模，注意到p^c是1e6范围内，既然是取模，那便知道一件事是：每隔p^c，单个数取模的结果就会循环一次，这样在循环节内部暴力求