组合数的计算

主讲人：邓哲也

组合数的计算

给定 n, m，计算

$$\binom{n}{m}$$

对 10^9+7 取模的结果。

组合数的计算

当 n，m 都比较小的时候

且需要频繁地调用组合数的时候

采用 Pascal 公式

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$$

组合数的计算

```
const int mo = 1e9 + 7;
int c[N][N];


for (int i = 0; i <= n; i ++){
      c[i][0] = c[i][i] = 1;
      for (int j = 1; j < i; j ++)
            c[i][j] = (c[i - 1][j - 1] + c[i][j - 1]) % mo;
}
```

组合数的计算

当 $n, m > 10000$的时候，就不能再使用 $O(n^2)$ 的做法了。

需要使用公式：

$$\binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

令 fact[n]=n!

```
fact[0] = 1;
for (int i = 1;i <= n;i ++)
      fact[i] = 1LL * fact[i - 1] * i % p;
```

组合数的计算

$$\binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

第一种方法：

令分母 k = 1LL * fact[m] * fact[n - m] % p

计算 k 模 p 的逆元，即 k^{p-2} MOD p

答案就是 1LL * fact[n] * k^{p-2} MOD p

这样需要一次快速幂，时间复杂度 O(log n)

组合数的计算

$$\binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

第二种方法：

我们先处理每个数关于 p 的逆元 inv[n]。

inv[1] = 1

现在假设我们要求 inv[i]，首先拿 p 除以 i，得到 k，余数是 r。

那么 $p = ki + r\ (r < i,\ 1 < i < p)$

两边都 MOD p，得到 $ki + r \equiv 0 \pmod p$

组合数的计算

$ki + r \equiv 0 \pmod p$

两边同时乘上 $i^{-1}r^{-1}$

得到 $kr^{-1} + i^{-1} \equiv 0 \pmod p$

$i^{-1} \equiv -kr^{-1} \pmod p$

$k = p / i,\ r = p \% i$

由于 $r < i$，所以算inv[i]的时候，inv[r]显然已经算好了

因此 inv[i] = (-p / i) * inv[p % i]

组合数的计算

在处理出了每个数关于 p 的逆元。

```
for (int i = 2;i <= n;i ++)
      inv[i] = 1LL * inv[i - 1] * inv[i] % p;
```

就可以得到 i! 关于 p 的逆元。

$$\binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

此时答案就是

```
    1LL * fact[n] * inv[m] % p * inv[n - m] % p
```

组合数的计算

给定 n, m，计算

$$\binom{n}{m}$$

对 1000003 取模的结果。

n， m <= 10^{18}

组合数的计算

Lucas定理：

$$\boldsymbol{Lucas(n,m,p) = Lucas\left(\frac{n}{p},\frac{m}{p},p\right) \times \binom{n \bmod p}{m \bmod p}}$$

递归即可。

运用预处理阶乘和阶乘的逆元的方法，可以在 $\log_p n$ 时间内求解。

下节课再见