组合数学基础

10circle

2025.2

1. Part 1 一些理论
   - 组合数学基础
   - 一些恒等式
   - 容斥原理
   - 生成函数
2. Part 2 怎么写代码?
   - 组合数的计算
3. Part 3 一些例题
   - 一些例题

OI-Wiki

- 组合数学基础

一些恒等式

- 组合数的对称性：$\binom{n}{m} = \binom{n}{n-m}$。
- 二项式定理：$(x+y)^n = \sum_{i=1}^{n} \binom{n}{i} x^i y^{n-i}$。
- 范德蒙德卷积：$\sum_{i=0}^{k} \binom{n}{i}\binom{m}{k-i} = \binom{n+m}{k}$，OI-Wiki。
- 一行组合数中奇数位置和偶数位置的和：$\sum_{i\text{为偶数},i\le n} \binom{n}{i} = \sum_{i\text{为奇数},i\le n} \binom{n}{i} = 2^{n-1}$。证明可以考虑对于前 $n-1$ 个任意一种选法，都可以调整最后一个选不选使得选了奇数个或选了偶数个。

组合数前缀和

纵向前缀和：$\sum_{n=k}^{m}\binom{n}{k}=\binom{m+1}{k+1}$，可以由递推式推导。正斜向的前缀和可以转化为纵向的。
多次询问组合数的横向前缀和做法十分复杂，网上有博客，可以自行搜索。有莫队做法和双 log 做法。一般说组合数前缀和也是指横向。

容斥原理

OI-Wiki

二项式反演

容斥原理的特例。

生成函数简介

组合数的计算

一般而言，由于组合数增长较快，都会在对某个模数取模的情况下计算组合数。

平方递推

$\binom{n}{m} = \binom{n-1}{m-1} + \binom{n-1}{m}$
组合意义证明：考虑 n 个元素的最后一个选不选。
该方法不要求模数是质数，可以 $O(nm)$ 预处理，$O(1)$ 查询。

平方递推

```
const int mod = 1e9 + 7, N = 5005;
int C[N][N];
void init() {
    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] + C[i - 1][j - 1]) % mod;
    }
}
// C[n][m] 就是组合数
```

模数是大质数时的 O(n) 预处理

由于模数是大质数（大于 n），所以可以认为在 n 以下的数都有逆元。于是预处理阶乘和阶乘逆元即可 $O(1)$ 计算。有一个一次循环处理阶乘、阶乘逆元、单个数的逆元的写法（也是我一般的写法），见下。原理可以看 OI-Wiki。

代码

```
ll f[N], fv[N], iv[N];
void init() {
    f[0] = fv[0] = iv[0] = f[1] = fv[1] = iv[1] = 1;
    for (int i = 2; i < N; ++i)
    f[i] = f[i - 1] * i % mod,
    iv[i] = (mod - mod / i) * iv[mod % i] % mod,
    fv[i] = fv[i - 1] * iv[i] % mod;
}
ll C(int n, int m) {
    return n < m || n < 0 || m < 0 ? 0 :
    f[n] * fv[m] % mod * fv[n - m] % mod;
}
```

当 m 很小的时候计算单个组合数

当 m 很小时，可以直接计算 $\frac{n(n-1)\cdots(n-m+1)}{m!}$。

- 如果模数是质数，可以计算 $m!$ 的逆元，然后乘上去，复杂度 $O(m)$。
- 如果模数是合数，就需要枚举 $1\sim m$ 的每个质因子，计算其在 $m!$ 中的出现次数，然后把在 $n, n-1, \cdots, n-m+1$ 中的部分除掉，复杂度 $O(m\log\log m)$。

某个质数 p 在 $n!$ 的唯一分解中的出现次数是 $\sum_{i=1}^{\infty}\lfloor\frac{n}{p^i}\rfloor$。

合数模数下的计算

```
ll C(ll n, int m) {
    vector<ll> a(m);
    ll k = n - m + 1;
    for (ll i = 0; i < m; ++i) a[i] = i + k;
    for (int p : prime) {
        if (p > m) break;
        ll vp = 0, pq = p;
        while (pq <= m) vp += m / pq, pq *= p;
        for (ll t = n / p * p; t >= n - m + 1; t -= p)
            while (vp && a[t - k] % p == 0)
                a[t - k] /= p, --vp;
    }
    ll ans = 1;
    for (ll i : a) ans = ans * (i % mod) % mod;
    return ans;
}
```

模小数时的 $O(\mathrm{mod})$ 算法

Lucas 定理 用于解决模数为质数时的问题。
下面的 exLucas 定理解决合数时的问题，exLucas 用处不大。

集合求和

组合数问题

前两道都是组合数的基础求法。

Cnoi 数学练习

对角线

画中漂流

糖果

硬币购物

消失之物

组合数问题

Lucas + 数位 dp 好题。