组合数学

2020-11-02 | 2022-10-07 | OI & ACM | 21
14k | 13 分钟

简介

排列和组合

1. 设 S 是有 k 种元素的多重集合，每种元素有无限多个，则 S 的 r 排列的个数是 k^r
2. 设 S 是有 k 种元素的多重集合，每一种元素有有限个，分别是 $n_1, n_2, \cdots, n_k$，设 $|S| = n = n_1 + n_2 + \cdots + n_k$，则 S 的 $|S|$ 排列数为 $\frac{n!}{n_1!n_2!\cdots n_k!}$，等价于 $\binom{n}{n_1}\binom{n-n_1}{n_2}\cdots\binom{n-n_1-n_2-\cdots-n_{k-1}}{n_k}$
3. 设 S 是有 k 种元素的多重集合，每一种元素有至少 r 个，则 S 的 r 组合为 $\binom{n+r-1}{r-1}$. 等价于方程 $x_1 + x_2 + \cdots + x_k = r$ 的非负整数解的个数
4. 设 S 是多重集合 $= \{n_1 \cdot a_1, n_2 \cdot a_2, \cdots, n_k \cdot a_k\}$，$n_1, n_2, \cdots, n_k$ 是非负整数，设 g_n 是 S 的 n 排列数，则 g 的指数型生成函数 $G(x) = F_{n_1}(x)F_{n2(x)}\cdots F_{n_k}(x)$，其中 $F_{n_k}(x) = \sum_{i=0}^{n_k}\frac{x^i}{i!}$
5. 设 S 是多重集合 $= \{n_1 \cdot a_1, n_2 \cdot a_2, \cdots, n_k \cdot a_k\}$，$n_1, n_2, \cdots, n_k$ 是非负整数，设 g_n 是 S 的 n 交错排列数，则 g 的指数型生成函数 $G(x) = F_{n_1}(x)F_{n2(x)}\cdots F_{n_k}(x)$，其中 $F_{n_k}(x) = \sum_{i=1}^{n_k}(-1)^{n_k-i}\binom{n_k-1}{i-1}\frac{x^i}{i!}$

球盒问题

1. 球不同，盒子不同，可以有空盒

   方案数 m^n

2. 球相同，盒子不同，不能有空盒

   相当于在 n 个小球的 $n-1$ 个空隙中插入 $m-1$ 个板子

   方案数为 $\binom{n-1}{m-1}$

3. 球相同，盒子不同，可以有空盒

   可以看做有 $n+m$ 个相同小球，盒子不同，不能有空盒的方案

方案数为 $\binom{n+m-1}{m-1}$

4.

卡特兰数

定义

n 个数依次进栈，出栈序列的个数为 H_n

常见公式

$H_n=\sum_{i=0}^{n-1}H_iH_{n-1-i}$，$H_0=1$

$H_n=H_{n-1}\frac{4n-2}{n+1}$

$H_n=\frac{\binom{2n}{n}}{n+1}$

$H_n=\binom{2n}{n}-\binom{2n}{n-1}$

二项式

定义

$\binom{n}{m}=\frac{n!}{m!(n-m)!},n,m\in N$，当 $m>n$ 时，$\binom{n}{m}=0$

我们尝试拓宽组合数的定义域，我们定义 n^{k} 为 n 的 k 次下降幂，$n^{k}=n\times(n-1)\times\cdots\times(n-k+1)$，那么

$$\binom{n}{m}=\frac{n^{m}}{m!}$$

这时候我们可以使组合数的上指标 n 为任意实数

二项式定理

$(a+b)^n=\sum_{i=0}^{n}\binom{n}{i}a^ib^{n-i},n\in N$

几个性质：

1. 从 n 个数里选奇数个数的方案等于从 n 个数中选偶数个数的方案等于 2^{n-1}

   证明：$(1-1)^n=\sum_{i=0}^{n}(-1)^i\binom{n}{i}$

牛顿二项式定理

设 r 是实数，对于所有满足 $0 \le |a| < |b|$ 的 a 和 b，有 $(a+b)^r = \sum_{i=0}^{\infty}\binom{r}{i}x^iy^{r-i}$

组合数恒等式

1. $\binom{n}{m} = \binom{n}{n-m}, n, m \in N$
2. $\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1}, n \in R, m \in N$
3. $m\binom{n}{m} = (n-m+1)\binom{n}{m-1}, n \in R, m \in N$
4. $m\binom{n}{m} = n\binom{n-1}{m-1}, n \in R, m \in N$
5. $(n-m)\binom{n}{m} = n\binom{n-1}{m}, n \in R, m \in N$
6. $\sum_{i=k}^{n}\binom{i}{k} = \binom{n+1}{k+1}, n, k \in N$

   证明:

   设有 $n+1$ 个物 品，标号为 $0 \sim n$，现在从中选取 $k+1$ 个物品，当选取的最大号码为 i 时，方案数为 $\binom{i}{k}$

   累加方案数即可

7. $\sum_{i=0}^{n}\binom{k+i}{i} = \binom{n+k+1}{n}, n, k \in N$

   证明:

   $\sum_{i=0}^{n}\binom{k+i}{i} = \sum_{i=0}^{n}\binom{k+i}{k} = \binom{k+n+1}{k+1} = \binom{k+n+1}{n}$

8. $\binom{n}{i}\binom{i}{k} = \binom{n-k}{i-k}\binom{n}{k}, n \in R, i, k \in N$
9. $\binom{-n}{k} = (-1)^k\binom{n+k-1}{k}, n \in R, k \in N$

   证明:

   把 $(-1)^k$ 移过去，然后展开就能证明

10. $\sum_{i=0}^{n}\binom{s}{i}\binom{t}{n-i} = \binom{s+t}{n}, s, t \in R, n \in N$

    推论：$\sum_{i=0}^{n}\binom{n}{i}^2 = \binom{2n}{n}$

    证明:

左边表示从 s 个男生中选 i 个人，从 t 个女生中选 $n-i$ 个人的方案数

右边表示从 $s+t$ 个人中选 n 个人的方案数

11. $\sum_{i=0}^{n} i\binom{n}{i} = n2^{n-1}$

    证明：

    $i\binom{n}{i} = n\binom{n-1}{i-1}$

12. $\sum_{i=0}^{n}\binom{n}{i}[i \equiv 1 \pmod 2] = \sum_{i=0}^{n}\binom{n}{i}[i \equiv 0 \pmod 2] = 2^{n-1}$

lucas定理

legendre公式和kummer定理

legendre公式

对于质数 p，函数 $v_p(n)$ 为 n 标准分解后 p 的次数，我们有 $v_p(n!) = \sum_{i=1}^{\infty}\lfloor\frac{n}{p^i}\rfloor$，令 $s_p(n)$ 表示 n 在 p 进制下的数位和，我们有 $v_p(n!) = \frac{n-s_p(n)}{p-1}$

kummer定理

$\binom{n+m}{m}$ 质因数分解后质数 p 的次数为 $n+m$ 在 p 进制下的进位次数

$v_p(\binom{n+m}{m}) = \frac{s_p(n)+s_p(m)-s_p(n+m)}{p-1}$

$v_p(\binom{n}{m_1,\cdots,m_k}) = \frac{\sum_{i=1}^{k}s_p(m_i)-s_p(n)}{p-1}$

求组合数

单个组合数

1. $O(n)$ 预处理阶乘和阶乘的逆元，单次 $O(1)$ 求

   需要注意的是阶乘逆元的处理 n 必须小于 p，要不然就全变成 0 了

```
ll inv[maxn], fac[maxn];
void init_inv() {
    fac[0] = 1; for (int i = 1; i <= n; ++i) fac[i] = fac[i - 1] * i % p;
    inv[n] = pow_mod(fac[n], p - 2); for (int i = n; ~i; --i) inv[i] = inv[i + 1] * (i + 1
}

ll C(int n, int m) { return n < m ? 0 : fac[n] * inv[m] % p * inv[n - m] % p; }
```

2. $O(n^2)$ 预处理

```
int C[maxn][maxn];
void init_C() {
    for (int i = 0; i <= n; ++i) C[i][0] = 1;
    for (int i = 1; i <= n; ++i)
        for (int j = 1; j <= i; ++j) C[i][j] = (C[i - 1][j] + C[i - 1][j - 1]) % p;
}
```

3. $O(n)$ 递推一行

```
int C[maxn]; ll inv[maxn];
void init_C(int n, int m) {
    inv[1] = 1; for (int i = 2; i <= n; ++i) inv[i] = -(p / i) * inv[p % i] % p;
    C[0] = 1;
    for (int i = 1; i <= m; ++i) C[i] = C[i - 1] * inv[i] % p * (n - i + 1) % p;
}
```

4. 利用 $lucas$ 定理，时间复杂度 $O(\log_p n + p^2)$

   大概就是将 n 和 m 在 p 进制下的每一位乘起来即可

```
ll Lucas(int n, int m) {
    if (!m) return 1;
    return C(n % p, m % p) * Lucas(n / p, m / p) % p;
}
```

组合数前缀和

令 $S(n,m)=\sum_{i=0}^{m}\binom{n}{i}$

$S(n,m+1)=S(n,m)+\binom{n}{m+1}$，$S(n+1,m)=2S(n,m)-\binom{n}{m}$

1. $O(n^2)$ 预处理

```
ll C[maxn][maxn], S[maxn][maxn];
void init_C(int n) {
    for (int i = 0; i <= n; ++i) C[i][0] = 1;
    for (int i = 1; i <= n; ++i)
        for (int j = 1; j <= i; ++j) C[i][j] = (C[i - 1][j] + C[i - 1][j - 1]) % p;
    for (int i = 0; i <= n; ++i) S[i][0] = 1;
    for (int i = 1; i <= n; ++i)
        for (int j = 1; j <= i; ++j) S[i][j] = (S[i][j - 1] + C[i][j]) % p;
}
```

2. 类似莫队做转移

```cpp
struct Csum {
    int l, r; ll inv2, sum;
    void init() { l = r = 1; sum = 2; inv2 = pow_mod(2, p - 2); }

    void move(int L, int R) {
        while (r < R) sum = (2 * sum - C(r++, l)) % p;
        while (l > L) sum = (sum - C(r, l--)) % p;
        while (r > R) sum = (sum + C(--r, l)) * inv2 % p;
        while (l < L) sum = (sum + C(r, ++l)) % p;
    }

    ll get() { return sum; }
} S;
```

3. 利用 $lucas$ 定理进行递归处理，时间复杂度 $O(\log_p^2 n + p^2)$

通过化简可以得到 $S(n,m) = S(\lfloor \frac{n}{p} \rfloor, \lfloor \frac{m}{p} \rfloor - 1) \times S(n \bmod p, p-1) + \binom{\lfloor \frac{n}{p} \rfloor}{\lfloor \frac{m}{p} \rfloor} \times S(n \bmod p, m \bmod p)$

化简过程如下，令 $m = kp + r$

$$
\begin{aligned}
S(n,m) &= \sum_{i=0}^{kp-1} \binom{n}{i} + \sum_{i=kp}^{kp+r} \binom{n}{i} \\
&= \sum_{i=0}^{kp-1} \binom{\lfloor \frac{n}{p} \rfloor}{\lfloor \frac{i}{p} \rfloor} \binom{n \bmod p}{i \bmod p} + \binom{\lfloor \frac{n}{p} \rfloor}{k} \sum_{i=0}^{r} \binom{n \bmod p}{i} \\
&= \sum_{i=0}^{k-1} \binom{\lfloor \frac{n}{p} \rfloor}{i} \sum_{i=0}^{p-1} \binom{n \bmod p}{i} + \binom{\lfloor \frac{n}{p} \rfloor}{k} S(n \bmod p, r) \\
&= S(\lfloor \frac{n}{p} \rfloor, k) \times S(n \bmod p, p-1) + \binom{\lfloor \frac{n}{p} \rfloor}{k} S(n \bmod p, r)
\end{aligned}
$$

```cpp
ll C[maxn][maxn], S[maxn][maxn];
void init_C(int n) {
    for (int i = 0; i <= n; ++i) C[i][0] = 1;
    for (int i = 1; i <= n; ++i)
        for (int j = 1; j <= i; ++j) C[i][j] = (C[i - 1][j] + C[i - 1][j - 1]) % p;
    for (int i = 0; i <= n; ++i) S[i][0] = 1;
    for (int i = 1; i <= n; ++i)
        for (int j = 1; j <= i; ++j) S[i][j] = (S[i][j - 1] + C[i][j]) % p;
}

ll Lucas(ll n, ll m) {
    if (!m) return 1;
    return C[n % p][m % p] * Lucas(n / p, m / p) % p;
}
```

```

ll calcS(ll n, ll m) {
    if (m < 0) return 0; m = min(n, m);
    if (n < p && m < p) return S[n][m];
    return (calcS(n / p, m / p - 1) * calcS(n % p, p - 1) + Lucas(n / p, m / p) * calcS(n
}
```

斯特林数

第一类斯特林数

定义

$\begin{bmatrix}n\\k\end{bmatrix}$，也可记作 $s(n,k)$，表示将 n 个两两不同的元素，划分为 k 个互不区分的非空轮换的方案数

递推式

$\begin{bmatrix}n\\k\end{bmatrix}=\begin{bmatrix}n-1\\k-1\end{bmatrix}+(n-1)\begin{bmatrix}n-1\\k\end{bmatrix}$，边界是 $\begin{bmatrix}n\\k\end{bmatrix}=[n=0]$

使用组合意义来证明：我们加入一个新元素时，将新元素单独放一个子集，有 $\begin{bmatrix}n-1\\k-1\end{bmatrix}$ 方案，将新元素放入一个现有子集，共有 $(n-1)\begin{bmatrix}n-1\\k\end{bmatrix}$ 方案

第二类斯特林数

定义

$\begin{Bmatrix}n\\k\end{Bmatrix}$，也可记作 $S(n,k)$，表示将 n 个两两不同的元素，划分成 k 个互不区分的非空子集的个数

递推式

$\begin{Bmatrix}n\\k\end{Bmatrix}=\begin{Bmatrix}n-1\\k-1\end{Bmatrix}+k\begin{Bmatrix}n-1\\k\end{Bmatrix}$，边界是 $\begin{Bmatrix}n\\0\end{Bmatrix}=[n=0]$

使用组合意义来证明：我们加入一个新元素时，将新元素单独放一个子集，有 $\begin{Bmatrix}n-1\\k-1\end{Bmatrix}$ 方案，将新元素放入一个现有子集，有 $k\begin{Bmatrix}n-1\\k\end{Bmatrix}$ 方案

通项公式

$$\begin{Bmatrix} n \\ k \end{Bmatrix} = \sum_{i=0}^{k} (-1)^{k-i} \frac{i^n}{i!(k-i)!}$$

考虑使用二项式反演来证明这个式子，我们令 f_k 表示将 n 个两两不同的元素划分为 k 个两两不同的集合，允许有空集的方案数，g_k 表示将 n 个两两不同的元素，划分为 k 个两两不同的非空集合的方案数

容易得到 $f_k = \sum_{i=0}^{k} \binom{k}{i} g_i = k^n$，根据二项式反演，$g_k = \sum_{i=0}^{k} (-1)^{k-i} \binom{k}{i} f_i$，且 $g_k = k! \begin{Bmatrix} n \\ k \end{Bmatrix}$

阶乘幂

上升幂

$$x^n = \prod_{i=0}^{n-1} (x+i) = \binom{x+n-1}{n} n!$$

下降幂

$$x^{\underline{n}} = \prod_{i=0}^{n-1} (x-i) = \binom{x}{n} n!$$

上升幂和下降幂的转换

$$x^{\underline{n}} = (-1)^n (-x)^n x^n = (-1)^n (-x)^{\underline{n}}$$

阶乘幂与斯特林数

$$x^n = \sum_{i=0}^{n} \begin{bmatrix} n \\ i \end{bmatrix} x^i$$
$$x^n = \sum_{i=0}^{n} \begin{Bmatrix} n \\ i \end{Bmatrix} x^{\underline{i}}$$
$$x^{\underline{n}} = \sum_{i=0}^{n} (-1)^{n-i} \begin{bmatrix} n \\ i \end{bmatrix} x^i$$

自然数幂和

我们令 $F_k(n) = \sum_{i=0}^{n} i^k$

我们利用 x^n 的与第一类斯特林数的关系，可以得到 $x^n = x^{\underline{n}} - \sum_{i=0}^{n-1} \begin{bmatrix} n \\ i \end{bmatrix} x^i$

$$
\begin{aligned}
F_k(n) &= \sum_{i=0}^{n} i^k \\
&= \sum_{i=0}^{n} (i^{\underline{k}} - \sum_{j=0}^{k-1} (-1)^{k-j} \begin{bmatrix} k \\ j \end{bmatrix} i^j) \\
&= \sum_{i=0}^{n} k! \binom{i}{k} - \sum_{j=0}^{k-1} (-1)^{k-j} \begin{bmatrix} k \\ j \end{bmatrix} \sum_{i=0}^{n} i^j \\
&= \frac{(i+1)^{\underline{k+1}}}{k+1} - \sum_{i=0}^{k-1} (-1)^{k-i} \begin{bmatrix} k \\ i \end{bmatrix} F_i(n)
\end{aligned}
$$

可以 $O(n^2)$ 递推，注意到不需要除法，因为连续 $k+1$ 个数中一定有一个数是 $k+1$ 的倍数

另外还可以做到 $O(k \log k)$，我们考虑利用第二类斯特林数进行化简，可以得到

$$
\begin{aligned}
F_k(n) &= \sum_{i=0}^{n} i^k \\
&= \sum_{i=0}^{n} \sum_{j=0}^{k} \begin{Bmatrix} k \\ j \end{Bmatrix} i^{\underline{j}} \\
&= \sum_{i=0}^{n} \sum_{j=0}^{k} \begin{Bmatrix} k \\ j \end{Bmatrix} \binom{i}{j} j! \\
&= \sum_{j=0}^{k} \begin{Bmatrix} k \\ j \end{Bmatrix} j! \sum_{i=0}^{n} \binom{i}{j} \\
&= \sum_{j=0}^{k} \begin{Bmatrix} k \\ j \end{Bmatrix} j! \binom{n+1}{j+1}
\end{aligned}
$$

用卷积预处理第二类斯特林数即可

固定 n，对于 $i \in [1, k]$，求 $F_i(n)$，我们考虑这个东西的生成函数

$$
\begin{aligned}
\sum_{i=0}^{k} \sum_{j=0}^{n} j^i \frac{x^i}{i!} &= \sum_{j=0}^{n} \sum_{i=0}^{k} \frac{(jx)^i}{i!} \\
&= \sum_{j=0}^{n} e^{jx} \\
&= \frac{e^{(n+1)x-1}}{e^x - 1}
\end{aligned}
$$

这个东西的分母常数项为 0，看起来不能直接求逆，但同时分子的常数项也为 0，所以可以同除 x 之后求逆，时间复杂度 $O(k \log k)$

求斯特林数

第一类斯特林数-列

简要题意：给定 n,k，求所有 $i\in[0,n]$，$\begin{bmatrix}i\\n\end{bmatrix}$，$n,k\le 2\times 10^5$

简要题解：我们构造 $k=1$ 的生成函数 $F(x)=\sum_{i=1}^{n}(i-1)!\frac{x^i}{i!}=\ln\frac{1}{1-x}$，注意到答案就是 $\frac{F^k(x)}{k!}$，除以 $k!$ 是因为圆排列是没有顺序的，时间复杂度 $O(n\log n)$

第一类斯特林数-行

简要题意：给定 n，求所有 $i\in[0,n]$，$\begin{bmatrix}n\\i\end{bmatrix}$，$n\le 2\times 10^5$

简要题解：我们考虑 $\begin{bmatrix}n\\k\end{bmatrix}$ 的生成函数 $F(x)=\sum_{i=0}^{n}\begin{bmatrix}n\\i\end{bmatrix}x^i=x^{\overline{n}}$，同时我们有 $x^{\overline{2n}}=x^{\overline{n}}(x+n)^{\overline{n}}$，即 $F_2n(x)=F_n(x)F_n(x+n)$，那么我们直接倍增即可，时间复杂度为 $O(n\log n)$

```
Poly solve(int n) {
    if (!n) return Poly { 1 };
    Poly res = solve(n / 2);
    res = Pol::operator*(res, Pol::Offset(res, n / 2));
    if (n & 1) {
        res.push_back(0);
        for (int i = n; ~i; --i)
            res[i] = ((i ? res[i - 1] : 0) + 1ll * res[i] * (n - 1)) % p;
    }
    return res;
}
```

第二类斯特林数-列

简要题意：给定 n,k，求所有 $i\in[0,n]$，$\begin{Bmatrix}i\\n\end{Bmatrix}$，$n,k\le 2\times 10^5$

简要题解：我们令 $F_k(x)=\sum_{i=0}^{\infty}\begin{Bmatrix}i\\k\end{Bmatrix}x^i$，根据第二类斯特林数递推公式 $\begin{Bmatrix}n\\k\end{Bmatrix}=\begin{Bmatrix}n-1\\k-1\end{Bmatrix}+k\begin{Bmatrix}n-1\\k\end{Bmatrix}$，我们有 $F_k(x)=xF_{k-1}(x)+kxF_k(x)$，$F_0(x)=1$

即 $F_k(x)=\frac{x^k}{\prod_{i=1}^{n}1-ix}$，下面的东西我们直接分治乘一下，最后求逆即可，时间复杂度 $O(n\log^2 n)$

第二类斯特林数-行

简要题意：给定 n，求所有 $i\in[0,n]$，$\begin{Bmatrix}n\\i\end{Bmatrix}$，$n\le 2\times 10^5$

简要题解：我们考虑第二类斯特林数通项公式 $\begin{Bmatrix}n\\k\end{Bmatrix}=\sum_{i=0}^{k}(-1)^{k-i}\frac{i^n}{i!(k-i)!}$，我们令 $A_i=\frac{(-1)^i}{i!},B_i=\frac{i^n}{i!}$，容易发现就是一个卷积的形式，时间复杂度 $O(n\log n)$

贝尔数

定义

贝尔数 B_n 表示将 n 个有标号的小球放到任意多个无标号盒子中的方案数

那么我们能够得到 $B_n=\sum_{i=0}^{n}\begin{Bmatrix}n\\i\end{Bmatrix}$

递推式

根据定义能够得到 $B_n=\sum_{i=0}^{n-1}\binom{n-1}{i}B_i$，边界为 $B_0=1$

证明我们考察第 n 个球的位置，如果它自己单独放，那么方案数就是 $\binom{n-1}{n-1}B_{n-1}$，如果它跟令一个求一起放，那么方案数就是 $\binom{n-1}{n-2}B_{n-2}$，以此类推即可

求贝尔数

$O(n^2)$ 递推

````
```


$O(n\log n)$ 多项式 $exp$

注意 $B_n$ 的递推式 $B_n=\sum_{i=0}^{n-1}\binom{n-1}{i}B_i$，令 $B(x)$ 为 $B$ 的 $EGF$，那么容易

```cpp
ll fac[maxn], inv[maxn]; Poly B;
void init(int n) {
    Poly A(n);
    fac[0] = 1; for (int i = 1; i <= n; ++i) fac[i] = fac[i - 1] * i % p;
    inv[n] = pow_mod(fac[n], p - 2); for (int i = n - 1; ~i; --i) inv[i] = inv[i + 1] * (i + 1
    for (int i = 0; i < n; ++i) A[i] = inv[i]; A[0] = 0;
    B = Pol::Exp(A);
    for (int i = 0; i < n; ++i) B[i] = B[i] * fac[i] % p;
}
````

贝尔数前 7 项：$1,1,2,5,15,52,203,877$

欧拉数(Eulerian Number)

定义

对于一个长度为 n 且有 k 个上升的排列定义为欧拉数 $\left\langle\begin{matrix}n\\k\end{matrix}\right\rangle$

递推式

$\left\langle{n \atop k}\right\rangle = (k+1)\left\langle{n-1 \atop k}\right\rangle + (n-k)\left\langle{n-1 \atop k-1}\right\rangle$，边界为 $\left\langle{n \atop 0}\right\rangle = 1$

一些恒等式

$$\begin{aligned} x^n &= \sum_{k=0}^{n}\left\langle{n \atop k}\right\rangle\binom{x+k}{n} \\ \left\langle{n \atop k}\right\rangle &= \sum_{i=0}^{k}(-1)^i\binom{n+1}{i}(k-i+1)^n \\ \left\{{n \atop k}\right\} &= \frac{1}{k!}\sum_{i=0}^{n}\left\langle{n \atop i}\right\rangle\binom{i}{n-k} \\ \left\langle{n \atop k}\right\rangle &= \sum_{i=0}^{n}(-1)^{n-i-k}i!\left\{{n \atop i}\right\}\binom{n-i}{k} \end{aligned}$$

求欧拉数

简要题意：给定 n，求所有 $i \in [0,n]$，$\left\langle{n \atop i}\right\rangle$，$n \le 2\times 10^5$

简要题解：我们考虑恒等式 $\left\langle{n \atop k}\right\rangle = \sum_{i=0}^{n}(-1)^{n-i-k}i!\left\{{n \atop i}\right\}\binom{n-i}{k}$，化简可以得到 $k!\left\langle{n \atop k}\right\rangle = \sum_{i=0}^{n}\frac{(-1)^{n-(i+k)}}{(n-(i+k))!}\left\{{n \atop i}\right\}i!(n-i)!$，我们令 $A_k = \left\{{n \atop k}\right\}k!(n-k)!, B_k = \frac{(-1)^{n-k}}{(n-k)!}$，容易发现这是一个减法卷积的式子，第二类斯特林数同样可以用卷积求，时间复杂度 $O(n\log n)$

二阶欧拉数

定义

对于有 k 个上升的多重集 $1,1,2,2,\cdots,n,n$ 的排列定义为 $\left\langle\left\langle{n \atop k}\right\rangle\right\rangle$

递推式

$\left\langle\left\langle{n \atop k}\right\rangle\right\rangle = (k+1)\left\langle\left\langle{n-1 \atop k}\right\rangle\right\rangle + (2n-k-1)\left\langle\left\langle{n-1 \atop k-1}\right\rangle\right\rangle$

一些恒等式

$$\left\{{x \atop x-n}\right\} = \sum_{i=0}^{n-1}\left\langle\left\langle{n \atop i}\right\rangle\right\rangle\binom{x+n-k-1}{2n}$$

斐波那契数

递推式

$F_i = F_{i-1} + F_{i-2}, F_0 = 0, F_1 = 1$

通项公式

$F_n = \frac{1}{\sqrt{5}}[(\frac{1+\sqrt{5}}{2})^n - (\frac{1-\sqrt{5}}{2})^n]$

性质

1. $\sum_{i=1}^{n} F_i = F_{n+1} - 1$
2. $\sum_{i=1}^{n} f_{2i-1} = f_{2n}$
3. $\sum_{i=1}^{n} f_{2i} = f_{2n+1} - 1$
4. $\sum_{i=1}^{n} f_i^2 = f_n f_{n+1}$
5. $f_{2n} = f_n^2 + 2f_{n-1}f_n$

整数拆分

定义

对于一个非负整数 n，它的一个拆分定义为若干个正整数 $a_1, \cdots, a_k$，满足：

1. $\sum_{i=1}^{k} a_i = n$
2. $a_1 \geq a_2 \geq \cdots \geq a_k$

求整数拆分

令 p_n 表示 n 的拆分方案数，我们考虑 p_n 的生成函数 $P(x)$

容易得到 $P(x) = \prod_{i=1}^{\infty} \frac{1}{1-x^i}$，然而我们看起来并不能直接把这个东西求出来

五边形定理

$$\begin{aligned}\prod_{i=1}^{\infty}(1-x^i) &= \sum_{i=-\infty}^{\infty}(-1)^i x^{i(3i-1)/2} \\ &= \sum_{i=0}^{\infty}(-1)^i x^{i(3i\pm1)/2} \\ &= 1 - x - x^2 + x^5 + x^7 - x^{12} - x^{15} + \cdots\end{aligned}$$

容易得到 $P(x)\prod_{i=1}^{\infty}(1-x^i)=1$，那么我们直接多项式求逆即可

例题

1. 简要题解：给定一个长度为 n 的数字和整数 k，求将这个数字划分成不超过 k 段的所有划分方法的价值和，定义一种划分方法的价值为它各段的价值和，一段数字的价值就是这个数字的值，例如：114514 的一种划分 $(114)(5)(14)$ 的总价值是 $114+5+14=133$

   $1\le k\le n\le 10^6$

   简要题解：我们考虑对于每个区间 $[l,r]$ 计算它的价值会被计算所少次

   为了方便设 $l>1\land r<n$，$s=l-1,t=n-r$

   容易得到次数就是 $\sum_{d=2}^{k-1}\sum_{i=1}^{d-1}\binom{s-1}{i-1}\binom{t-1}{d-i-1}=\sum_{d=2}^{k-1}\binom{s+t-2}{d-2}=\sum_{d=0}^{k-3}\binom{s+t-2}{d}$，这是一个组合数前缀和的形式

   容易发现，这个东西只跟 $[l,r]$ 的长度有关，我们令 f_i 表示长度为 i 的区间的和，那么可以得到随着 i 的增加，这个组合数的上指标在减少，且总共只会移动 $O(n)$，我们可以 $O(n)$ 维护它的变化

   另外前缀和后缀以及 $k\le 2$ 的情况要单独考虑，时间复杂度 $O(n)$

------------- 本文结束 感谢阅读 -------------

Tech # 组合数学

< CF 1422D Returning Home   UVA11987 Almost Union-Find >