组合数学

1. 排列与组合

1.1 排列

（1）在没有其他条件的情况下，从 n 个不同元素中选取 r 个不同的元素的排列数为$A_n^r = \frac{n!}{(n-r)!}$，当 $r > n$ 时，A_n^r=0

（2）在 n 个不同元素中选取 r 个元素的圆排列的个数为$\frac{A_n^r}{r} = \frac{n!}{r \cdot (n-r)!}$

1.2 组合

（1）在在没有其他条件的情况下，从 n 个不同元素中选取r个不同的元素的排列数为$C_n^r = \frac{n!}{r!(n-r)!}$，当 $r > n$ 时，C_n^r=0

1.3 多重集的排列

（1）设元素$a_1, a_1, \cdots, a_n$互不相同，从无限多重集$\{\infty \cdot a_1, \infty \cdot a_2, \cdots \infty \cdot a_n\}$中选取 r 个元素的排列数为n^r

（2）设元素$a_1, a_1, \cdots, a_n$互不相同，从有限多重集$\{K_1 \cdot a_1, K_2 \cdot a_2, \cdots K_n \cdot a_n\}$中选取 r 个元素，当$K_1 \geq r, K_2 \geq r, \cdots, K_n \geq r$,时，排列数为$n^r$

（3）设元素$a_1, a_1, \cdots, a_n$互不相同，有限多重集$\{K_1 \cdot a_1, K_2 \cdot a_2, \cdots K_n \cdot a_n\}$中元素的全排列数为$\frac{(K_1+K_2+\cdots+K_n)!}{K_1!K_2!\cdots K_n!}$

（4）设元素$a_1, a_1, \cdots, a_n$互不相同，从有限多重集$\{K_1 \cdot a_1, K_2 \cdot a_2, \cdots K_n \cdot a_n\}$中选取 r 个元素，至少存在一个$K_i < r$时，排列数为$\sum_{k_1+k_2+\cdots+k_n=r} \frac{r!}{k_1!k_2!\cdots k_n!}$

1.4 多重集的组合

（1）设元素$a_1, a_1, \cdots, a_n$互不相同，从无限多重集$\{\infty \cdot a_1, \infty \cdot a_2, \cdots \infty \cdot a_n\}$中选取 r 个元素的组合数为C_{n+r-1}^r

（2）设元素$a_1, a_1, \cdots, a_n$互不相同，从有限多重集$\{K_1 \cdot a_1, K_2 \cdot a_2, \cdots K_n \cdot a_n\}$中选取 r 个元素，当$K_1 \geq r, K_2 \geq r, \cdots, K_n \geq r$,时，组合数为$C_{n+r-1}^r$

（3）设元素$a_1, a_1, \cdots, a_n$互不相同，从有限多重集$\{K_1 \cdot a_1, K_2 \cdot a_2, \cdots K_n \cdot a_n\}$中选取 r 个元素，至少存在一个$K_i < r$时，组合数通过容斥定理或生成函数可以求得

1.5 二项式定理

（1）$(a+b)^n = a^n + C_n^1 a^{n-1} b + C_n^2 a^{n-2} b + \cdots + b^n = \sum_{i=0}^n a^{n-i} b^i$

1.6 鸽巢(抽屉)原理

（1）有 $n+1$ 个物品放到 n 个抽屉中，有一个抽屉中至少会有两个物品

1.7* 组合数浅谈

1.7.1 组合数公式

下面是几个组合数公式，可以结合杨辉三角理解

1. $C_n^k = C_{n-1}^{k-1} + C_{n-1}^k$ （杨辉恒等式）
2. $C_n^k = C_n^{n-k}$ （杨辉三角对称性）
3. $\sum_{i=0}^n C_n^i = 2^n$ （单行和）
4. $\sum_{i=0}^n (C_n^i)^2 = C_{2n}^n$ （单行平方和）
5. $\sum_{i=0}^n C_{k+i}^k = C_{n+k+1}^{k+1}$ （60°斜行和）
6. $F_n = \begin{cases} \sum_{i=0}^{n/2-1} C_{n/2+i}^{2i+1}, n \equiv 0(mod\ 2) \\ \sum_{i=0}^{(n-1)/2} C_{(n-1)/2+i}^{2i}, n \equiv 1(mod\ 2) \end{cases}$ （30°斜行和等于斐波那契数列）
7. $C_n^i = \frac{n-i+1}{i} C_n^{i-1}$ （杨辉三角的一行可以递推）

1.7.2 组合数的求法

在ACM竞赛中，我们常常需要计算 $C_n^m \% p$ ，可以参考下面几种方法

1. 如果 n, m 很小（不超过50），可以用C++的库函数 `double tgamma(double x)` ，这是一个欧拉积分

$$\Gamma(s) = \int_0^{+\infty} x^{s-1} e^{-x} dx$$

在整数点处的取值满足

$$\Gamma(n+1) = n!$$

因此代码可以这么写

```
ll C(ll n,ll m){
    return (ll)round(tgamma(n+1)/tgamma(m+1)/tgamma(n-m+1));
}
```

效率并不高，但是对于追求手速来说足够了

2. 如果 n, m 不大，可以开 $O(n^2)$ 的空间，可以利用杨辉恒等式来预处理组合数表

```
const ll mo=1e9+7;
ll C[1005][1005];
void getC(int n){
    for(int i=0;i<=n;i++){
        for(int j=0;j<=i;j++){
            if(j==0 || j==i)
                C[i][j]=1;
            else
                C[i][j]=(C[i-1][j-1]+C[i-1][j])%mo;
        }
    }
}
```

3. 如果 n, m 比较大，可以开 $O(n)$ 的空间，可以利用前文所述的逆元来求解，当然，要保证 p 是素数

```
const ll mo=1e9+7;
ll C(ll n,ll m){
    static ll M=0,inv[N],mul[N],invMul[N];
    while(M<=n){
        if(M){
```

```
            inv[M]=M==1?1:(mo-mo/M)*inv[mo%M]%mo;
            mul[M]=mul[M-1]*M%mo;
            invMul[M]=invMul[M-1]*inv[M]%mo;
        }
        else mul[M]=1,invMul[M]=1;
        M++;
    }
    return mul[n]*invMul[m]%mo*invMul[n-m]%mo;
}
```

上面的代码中用 $O(n)$ 的复杂度处理了 $[1,n]$ 的逆元，处理 Q 次 $n,m\le N$ 的询问的总复杂度为 $O(N+Q)$

4. 如果 n,m 更大， p 是素数，可以用Lucas定理来求解

Lucas定理 若 p 是素数，则

$$C_n^m=\prod_{i=0}^{k}C_{n_i}^{m_i}\,(mod\,p)$$

其中

$$n=\sum_{i=0}^{k}n_ip^i$$

$$m=\sum_{i=0}^{k}m_ip^i$$

即将 n,m 表示成 p 进制形式

推论

$$C_n^m\equiv\chi(n\&m=m)(mod\,2)$$

```
ll Lucas(ll n,ll m,ll p){
    ll ans=1;
    while(n|m)ans=ans*C(n%P,m%P)%P,n/=P,m/=P;
    return ans;
}
```

5. 如果 n 固定，可以利用上面的公式7对 m 进行递推

2. 容斥原理

2.1 容斥原理

（1）设$A_1,A_2,\cdots,A_n$是集合 S 的子集，表示以集合 S 代表可能发生的事件中的 n 个子事件，$|A_i|$表示子事件A_i发生的个数$(0\le i\le n)$，则有

$$|A_1\cup A_2\cup A_3\cup\cdots\cup A_n|=\sum_{i=1}^{n}|A_i|-\sum_{1\le i<j\le n}|A_i\cap A_j|+\cdots+(-1)^{n-1}\left|A_1\cap A_2\cap A_3\cap\cdots\cap A_n\right|$$

$$\left|\bar{A_1}\cap\bar{A_2}\cap\bar{A_3}\cap\cdots\cap\bar{A_n}\right|=S-|A_1\cup A_2\cup A_3\cup\cdots\cup A_n|$$

2.2 错排问题

（1）设D_n表示$1,2,3,\cdots,n$这n个数的一个排列的错排个数，有

$$D_n=n!\left[1-\frac{1}{1!}+\frac{1}{2!}-\frac{1}{3!}+\cdots+(-1)^n\frac{1}{n!}\right],D_1=0,D_2=1$$

$$D_n=(n-1)[D_{n-1}+D_{n-2}],n>2$$

$$D_n=nD_{n-1}+(-1)^n$$

2.3 带有禁位的错排问题

（1）n个元素$a_1,a_2,a_3,\cdots,a_n$带有禁位$X_1,X_2,X_3,\cdots,X_n$的错排数为

$P\left(X_1,X_2,X_3,\cdots,X_n\right)=n!-r_1\left(n-1\right)!+r_2\left(n-2\right)!-\cdots+(-1)^k r_k\left(n-k\right)!+\cdots+(-1)^n r_n$

式中r_k表示有 k 个元素在禁位上的个数

3. 特殊计数

3.1 斐波那契数列

（1）满足递推方程$F_i=F_{i-1}+F_{i-2},i\geq 3;F_1=F_2=1$，的数列$\{F_n\}$称为斐波那契数列，$F_n$为斐波那契数。

（2）斐波那契数列的通项公式为$F_n=\frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n-\left(\frac{1-\sqrt{5}}{2}\right)^n\right]$

（3）$F_n\equiv 276601605(691504013^n-308495997^n)(mod\ (10^9+9))$

3.2 Catalan数

（1）Catalan数满足递推方程$C_n=\sum_{k=0}^{n-1}C_kC_{n-k-1},n\geq 2;C_0=C_1=1$

（2）前几个Catalan数为1,1,2,5,14,42,132,429,1430,4862

（3）Catalan数的通项公式为$C_n=\frac{C_{2n}^n}{n+1}=C_{2n}^n-C_{2n}^{n-1}$

（4）Catalan数的另一个递推公式为$C_n=\frac{4n-2}{n+1}C_{n-1}$

3.3 第一类Stirling数

（1）多项式$[x]_n=x\left(x-1\right)\left(x-2\right)\cdots\left(x-n+1\right)$中常数项和$x,x^2,x^3,\cdots,x^n$的系数称为第一类Stirling数，记为$S_1\left(n,k\right),k=0,2,\cdots,n$

（2）第一类Stirling数满足
$S_1\left(n,n\right)=1;S_1\left(n,0\right)=0;S_1\left(n,n-k\right)=(-1)^n M_k^n,k=1,2,\cdots,n-1$

式中M_k^n表示$\{1,2,\cdots,n-1\}$中任意k个不同的自然数乘积之和。

（3）第一类Stirling数满足递归关系$\begin{cases}S_1\left(n+1,k\right)=S_1\left(n,k-1\right)-nS_1\left(n,k\right),n\geq 0,k>0\\ S_1\left(n,n\right)=1,S_1\left(n,0\right)=0\end{cases}$

3.4 第二类Stirling数

（1）多项式$[x]_n=x\left(x-1\right)\left(x-2\right)\cdots\left(x-n+1\right)$，$x^n=\sum_{k=0}^{n}S_2\left(n,k\right)[x]_k$，称$S_2\left(n,k\right)$为第二类Stirling数。

（2）第二类Stirling数满足$S_2\left(n,k\right)=\frac{1}{k!}\sum_{i=0}^{k-1}(-1)^i C_k^i(k-i)^n$

（3）第二类Stirling数满足递归关系$\begin{cases} S_2\left(n+1,k\right)=S_2\left(n,k-1\right)+kS_2\left(n,k\right),n\geq 0,k\geq 1 \\ S_2\left(0,0\right)=1,S_2\left(n,0\right)=0 \end{cases}$

（4）第二类Stirling数可以用卷积的方法求，根据（2）得 $S_2(n,k)=\sum_{i=0}^{k-1}\frac{(-1)^i}{i!}\frac{(k-i)^n}{(k-i)!}$ ，对 $a_i=\frac{(-1)^i}{i!}$ 与 $b_i=\frac{i^n}{i!}$ 卷积即可

3.5 分拆数

（1）称正整数n分解为r个正整数和的个数为n分解成r的分拆数，记为$P_r\left(n\right)$

（2）$P_1\left(n\right)$=1；$P_n\left(n\right)$=1；$P_{n-1}\left(n\right)$=1；$P_{n-2}\left(n\right)$=2；$P_{n-3}\left(n\right)$=3

（3）$P_2\left(n\right)=\left\lceil\frac{n-1}{2}\right\rceil,n\geq 2$

（4）$P_r\left(n\right)=P_1\left(n-r\right)+P_2\left(n-r\right)+\cdots+P_r\left(n-r\right)$

3.6 分装问题

将n个球放入r个盒子称为分装问题

（1）相同球和相同盒子，n≥r

①没有空盒子：$P_r\left(n-r\right)$

②可以有空盒子：$\sum_{k=1}^{r}P_k\left(n\right)$

（2）相同球和不同盒子

①没有空盒子：C_{n-1}^{r-1}

②可以有空盒子：C_{n+r-1}^{n}

（3）不同球和相同盒子

①没有空盒子：$S_2\left(n,r\right)$

②可以有空盒子：$\sum_{k=1}^{r}S_2\left(n,k\right)$

（4）不同球和不同盒子

①没有空盒子：$r!S_2\left(n,r\right)$

②可以有空盒子：r^n

4. 生成函数

4.1 生成函数

对于一个序列$\{a_i\}$，如果这个序列的每一项a_i是幂函数$\sum_{i=0}^{\infty}a_ix^i$中各不同的$x^i$的系数，称幂函数$\sum_{i=0}^{\infty}a_ix^i$是序列$\{a_i\}$的生成函数，也称为母函数。

4.2 指数生成函数

对于一个序列$\{a_i\}$，如果这个序列的每一项a_i是幂函数$\sum_{i=0}^{\infty}a_i\frac{x^i}{i!}$中各不同的$x^i$的系数，称幂函数$\sum_{i=0}^{\infty}a_i\frac{x^i}{i!}$是序列$\{a_i\}$的指数生成函数。

4.3 利用生成函数求有限多重集的组合

设元素$a_1, a_1, \cdots, a_n$互不相同，从有限多重集$\{K_1 \cdot a_1, K_2 \cdot a_2, \cdots K_n \cdot a_n\}$中选取r个元素，至少存在一个$K_i < r$时，求其组合。

令$F_i = 1 + x + x^2 + \cdots + x^{K_i}, i = 1, 2, \cdots, n$，$\prod_{i=1}^{n} F_i$中$x^r$的系数即为所求，这里可能需要快速傅里叶变换

4.4 利用指数生成函数求有限多重集的排列

设元素$a_1, a_1, \cdots, a_n$互不相同，从有限多重集$\{K_1 \cdot a_1, K_2 \cdot a_2, \cdots K_n \cdot a_n\}$中选取r个元素，至少存在一个$K_i < r$时，求其排列。

令$F_i = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots + \frac{x^{K_i}}{K_i!}, i = 1, 2, \cdots, n$，$\prod_{i=1}^{n} F_i$中$x^r$的系数$\sum_{k_1+k_2+\cdots+k_n=r} \frac{r!}{k_1!k_2!\cdots k_n!}$即为所求

5. 线性递推数列

5.1线性递推方程

$$F_n - b_1 F_{n-1} - b_2 F_{n-2} - \ldots - b_k F_{n-k} = 0$$

其通项公式为

$$F_n = c_1 q_1^n + c_2 q_2^n + \ldots + c_k q_k^n$$

其中$q_1, q_2, \ldots, q_k$是特征方程

$$q^k - b_1 q^{k-1} - b_2 q^{k-2} - \ldots - b_k = 0$$

的根

$c_1, c_2, \ldots, c_k$是常数，由初值条件决定

5.2 非线性递推方程

$$F_n - b_1 F_{n-1} - b_2 F_{n-2} - \ldots - b_k F_{n-k} = S(n)$$

其通项公式为

$$F_n = c_1 q_1^n + c_2 q_2^n + \ldots + c_k q_k^n + f_n$$

其中$q_1, q_2, \ldots, q_k, c_1, c_2, \ldots, c_k$与上文相同， f_n为一特解

注：这只是给出了递推方程的一种求通解的理论方法，实际上高次多项式求根以及求递推方程的特解往往是很困难的，在ACM中，若要计算线性递推数列第n项的值，常用矩阵快速幂求解

6. Polya计数

6.1 Burnside 定理

非等价着色数等于置换群中保持不变的着色的平均数

$$N(G, C) = \frac{1}{|G|} \sum_{f \in G} C(f)$$

6.2 Polya计数公式

$$|C(f)| = k^{\# f}$$