

FOI2023 "简单" 数学

张志心

Zhejiang University

2023 年 7 月 14 日

关于今天的上课大纲：

排列组合、卡特兰数、斐波那契数、快速幂、欧几里得算法同余式、线性筛、欧拉定理与欧拉函数、费马小定理

① 快速幂

② 组合数学

③ 特殊的数

④ 质数，合数，约数，倍数

⑤ 同余方程

⑥ 欧拉函数

若 $n = 2^{k_1} + 2^{k_2} + \cdots + 2^{k_m}, k_1 < k_2 < \cdots < k_m$;

若 $n = 2^{k_1} + 2^{k_2} + \cdots + 2^{k_m}, k_1 < k_2 < \cdots < k_m$;
 $x^n = \prod_{i=1}^m x^{2^{k_i}}.$

若 $n = 2^{k_1} + 2^{k_2} + \cdots + 2^{k_m}, k_1 < k_2 < \cdots < k_m$;

$$x^n = \prod_{i=1}^m x^{2^{k_i}}.$$

把 x 不断变成 x^2 , 得到 x^{2^k} 的形式, 把需要的项乘入答案中。

直接看代码：

```
1 const int Mod = 1e9+7;
2 int Mul(int x, int y) {return (1ll * x * y) % Mod;}
3 int Add(int x, int y) {return (x + y) % Mod;}
4 int qpow(int x, long long y) {
5     int ret = 1;
6     for(;y;y>>=1, x=Mul(x,x))
7         if(y&1) ret = Mul(ret, x);
8     return ret;
9 }
```

① 快速幂

② 组合数学
排列组合
容斥原理

③ 特殊的数

④ 质数，合数，约数，倍数

⑤ 同余方程

⑥ 欧拉函数

① 快速幂

② 组合数学
排列组合
容斥原理

③ 特殊的数

④ 质数，合数，约数，倍数

⑤ 同余方程

⑥ 欧拉函数

加法原理:

如果完成一件事有 n 个方法, 第 i 种方法有 a_i 种方案完成, 则总共有 $\sum_{i=1}^n a_i$ 种方案完成这件事。

加法原理：

如果完成一件事有 n 个方法，第 i 种方法有 a_i 种方案完成，则总共有 $\sum_{i=1}^n a_i$ 种方案完成这件事。

乘法原理：

如果完成一件事有 n 个步骤，第 i 个步骤有 a_i 种方案完成，则总共有 $\prod_{i=1}^n a_i$ 种方案完成这件事。

- n 个不同的物品排成一行，求方案数。

- n 个不同的物品排成一行，求方案数。
- 选出的第一个物品有 n 种方案，第二个物品有 $n - 1$ 种方案……第 n 个物品有 1 种方案。

- n 个不同的物品排成一行，求方案数。
- 选出的第一个物品有 n 种方案，第二个物品有 $n - 1$ 种方案……第 n 个物品有 1 种方案。
- 因此，总方案数为 $n(n - 1) \cdots 1 = n!$ 。

- n 个不同的物品，选出 m 个排成一行， $m \leq n$ ，求方案数。

- n 个不同的物品，选出 m 个排成一行， $m \leq n$ ，求方案数。
- 选出的第一个物品有 n 种方案，第二个物品有 $n - 1$ 种方案……第 m 个物品有 $n - m + 1$ 种方案。

- n 个不同的物品，选出 m 个排成一行， $m \leq n$ ，求方案数。
- 选出的第一个物品有 n 种方案，第二个物品有 $n - 1$ 种方案……第 m 个物品有 $n - m + 1$ 种方案。
- 因此，总方案数为 $n(n - 1) \cdots (n - m + 1) = \frac{n!}{(n - m)!}$

- n 个不同的物品，选出 m 个排成一行， $m \leq n$ ，求方案数。
- 选出的第一个物品有 n 种方案，第二个物品有 $n - 1$ 种方案……第 m 个物品有 $n - m + 1$ 种方案。
- 因此，总方案数为 $n(n - 1) \cdots (n - m + 1) = \frac{n!}{(n - m)!}$
- 记该方案数为 P_n^m 或 A_n^m 。

- n 个不同的物品，选出 m 个排成一行， $m \leq n$ ，求方案数。
- 选出的第一个物品有 n 种方案，第二个物品有 $n - 1$ 种方案……第 m 个物品有 $n - m + 1$ 种方案。
- 因此，总方案数为 $n(n - 1) \cdots (n - m + 1) = \frac{n!}{(n - m)!}$
- 记该方案数为 P_n^m 或 A_n^m 。
- 特别地，若 $m = n$ ，则排列方案数为 $P_n^n = n!$ 。

- n 个不同的物品，选出 m 个排成一行， $m \leq n$ ，求方案数。
- 选出的第一个物品有 n 种方案，第二个物品有 $n - 1$ 种方案……第 m 个物品有 $n - m + 1$ 种方案。
- 因此，总方案数为 $n(n - 1) \cdots (n - m + 1) = \frac{n!}{(n - m)!}$
- 记该方案数为 P_n^m 或 A_n^m 。
- 特别地，若 $m = n$ ，则排列方案数为 $P_n^n = n!$ 。
- 例如 $P_5^3 = 5 \times 4 \times 3 = \frac{5!}{3!} = 60$ 。

- n 个不同的物品，选出 m 个， $m \leq n$ ，求方案数。顺序不同的算作一种。

- n 个不同的物品，选出 m 个， $m \leq n$ ，求方案数。顺序不同的算作一种。
- 因为顺序不同的算一种，所以方案数就是 $\frac{P_n^m}{m!} = \frac{n!}{m!(n-m)!}$ 。
- 记该方案数为 C_n^m 或 $\binom{n}{m}$ 。

- n 个不同的物品，选出 m 个， $m \leq n$ ，求方案数。顺序不同的算作一种。
- 因为顺序不同的算一种，所以方案数就是 $\frac{P_n^m}{m!} = \frac{n!}{m!(n-m)!}$ 。
- 记该方案数为 C_n^m 或 $\binom{n}{m}$ 。
- 由于阶乘本身非常大，所以在遇到需要用排列数和组合数的问题时，一般会要求答案对某个大质数（指比题中条件大 2 个数量级以上）取模。此时我们预处理出阶乘和阶乘的逆元，即可 $O(1)$ 求出所有组合数。

- n 个不同的物品，选出 m 个， $m \leq n$ ，求方案数。顺序不同的算作一种。
- 因为顺序不同的算一种，所以方案数就是 $\frac{P_n^m}{m!} = \frac{n!}{m!(n-m)!}$ 。
- 记该方案数为 C_n^m 或 $\binom{n}{m}$ 。
- 由于阶乘本身非常大，所以在遇到需要用排列数和组合数的问题时，一般会要求答案对某个大质数（指比题中条件大 2 个数量级以上）取模。此时我们预处理出阶乘和阶乘的逆元，即可 $O(1)$ 求出所有组合数。
- 例如 $\binom{7}{3} = \frac{7 \times 6 \times 5}{1 \times 2 \times 3} = \frac{7!}{3!4!} = 35$ 。

- n 个不同的物品，选出 m 个， $m \leq n$ ，求方案数。顺序不同的算作一种。
- 因为顺序不同的算一种，所以方案数就是 $\frac{P_n^m}{m!} = \frac{n!}{m!(n-m)!}$ 。
- 记该方案数为 C_n^m 或 $\binom{n}{m}$ 。
- 由于阶乘本身非常大，所以在遇到需要用排列数和组合数的问题时，一般会要求答案对某个大质数（指比题中条件大 2 个数量级以上）取模。此时我们预处理出阶乘和阶乘的逆元，即可 $O(1)$ 求出所有组合数。
- 例如 $\binom{7}{3} = \frac{7 \times 6 \times 5}{1 \times 2 \times 3} = \frac{7!}{3!4!} = 35$ 。
- 因为阶乘逆元如果全部现算，复杂度会多一个 \log ，这在 10^7 的范围下会超时。因此我们一般倒序求阶乘逆元，这样就只需求一遍 $n!$ 的逆元了。

求组合数代码

```
1 ll fac[N],inf[N];
2 void init(int n){
3     fac[0]=1;
4     for(int i=1;i<=n;++i)
5         fac[i]=fac[i-1]*i%mod;
6     inf[n]=inv(fac[n]);
7     for(int i=n-1;~i;--i)
8         inf[i]=inf[i+1]*(i+1)%mod;
9 }
10 ll C(int n,int m){
11     return fac[n]*inf[m]%mod*inf[n-m]%mod;
12 }
```

- 7 个人站成一排，其中 A, B, C 三个人必须站在一起， X, Y 两个人必须站在两侧，求总方案数。

- 7 个人站成一排，其中 A, B, C 三个人必须站在一起， X, Y 两个人必须站在两侧，求总方案数。

$$2! \times 3! \times 3! = 72$$

- 7 个人站成一排，其中 A, B, C 三个人必须站在一起， X, Y 两个人必须站在两侧，求总方案数。
 $2! \times 3! \times 3! = 72$
- 7 个人站成一排， A, B, C 不能相邻，求总方案数。

- 7 个人站成一排，其中 A, B, C 三个人必须站在一起， X, Y 两个人必须站在两侧，求总方案数。

$$2! \times 3! \times 3! = 72$$

- 7 个人站成一排， A, B, C 不能相邻，求总方案数。

$$4! \times P_5^3 = 24 \times 60 = 1440$$

- 7 个人站成一排，其中 A, B, C 三个人必须站在一起， X, Y 两个人必须站在两侧，求总方案数。
 $2! \times 3! \times 3! = 72$
- 7 个人站成一排， A, B, C 不能相邻，求总方案数。
 $4! \times P_5^3 = 24 \times 60 = 1440$
- 求方程 $x_1 + x_2 + x_3 + x_4 = 12$ 的正整数解/非负整数解组数。

- 7 个人站成一排，其中 A, B, C 三个人必须站在一起， X, Y 两个人必须站在两侧，求总方案数。

$$2! \times 3! \times 3! = 72$$

- 7 个人站成一排， A, B, C 不能相邻，求总方案数。

$$4! \times P_5^3 = 24 \times 60 = 1440$$

- 求方程 $x_1 + x_2 + x_3 + x_4 = 12$ 的正整数解/非负整数解组数。

算法一： $\binom{11}{3} = 165$.

$$\binom{11}{3} + \binom{4}{1} \binom{11}{2} + \binom{4}{2} \binom{11}{1} + \binom{4}{3} \binom{11}{0} = 165 + 220 + 66 + 4 = 455$$

- 7 个人站成一排，其中 A, B, C 三个人必须站在一起， X, Y 两个人必须站在两侧，求总方案数。

$$2! \times 3! \times 3! = 72$$

- 7 个人站成一排， A, B, C 不能相邻，求总方案数。

$$4! \times P_5^3 = 24 \times 60 = 1440$$

- 求方程 $x_1 + x_2 + x_3 + x_4 = 12$ 的正整数解/非负整数解组数。

算法一： $\binom{11}{3} = 165$.

$$\binom{11}{3} + \binom{4}{1} \binom{11}{2} + \binom{4}{2} \binom{11}{1} + \binom{4}{3} \binom{11}{0} = 165 + 220 + 66 + 4 = 455$$

算法二： $(x_1 + 1) + (x_2 + 1) + (x_3 + 1) + (x_4 + 1) = 16$

$$\binom{15}{3} = 455$$

- 8 个人报名节目，其中 3 个只会唱歌，3 个只会跳舞，2 个又会唱又会跳，选 4 个人参加演出，2 人唱歌 2 人跳舞，求方案数。

- 8 个人报名节目，其中 3 个只会唱歌，3 个只会跳舞，2 个又会唱又会跳，选 4 个人参加演出，2 人唱歌 2 人跳舞，求方案数。

$$\binom{3}{0} \binom{2}{2} \binom{3}{2} + \binom{3}{1} \binom{2}{1} \binom{4}{2} + \binom{3}{2} \binom{2}{0} \binom{5}{2} = 3 + 36 + 30 = 69.$$

- 这是组合数的前 9 行，从 $\binom{0}{0}$ 到 $\binom{8}{8}$ 。

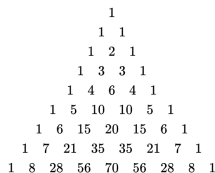


图 1: 杨辉三角

- 这是组合数的前 9 行，从 $\binom{0}{0}$ 到 $\binom{8}{8}$ 。
- 容易发现下面的数恰好是上面两个数的和，即

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

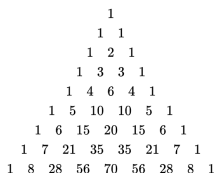


图 1: 杨辉三角

- 这是组合数的前 9 行，从 $\binom{0}{0}$ 到 $\binom{8}{8}$ 。
- 容易发现下面的数恰好是上面两个数的和，即

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

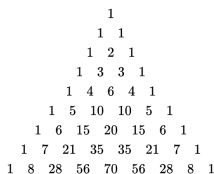


图 1: 杨辉三角

- 考虑实际意义。如果最后一个物品选了，那么其余部分的方案数为 $\binom{n-1}{m-1}$ ；没选，其余部分方案数为 $\binom{n-1}{m}$ 。因此总方案数就是二者相加。

- 这是组合数的前 9 行，从 $\binom{0}{0}$ 到 $\binom{8}{8}$ 。
- 容易发现下面的数恰好是上面两个数的和，即

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

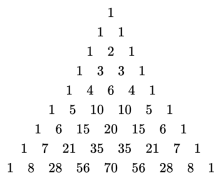


图 1: 杨辉三角

- 考虑实际意义。如果最后一个物品选了，那么其余部分的方案数为 $\binom{n-1}{m-1}$ ；没选，其余部分方案数为 $\binom{n-1}{m}$ 。因此总方案数就是二者相加。
- 根据上面的递推公式，我们可以 $O(n^2)$ 求出所有的 $\binom{n}{m}$ 。

求 $i \leq n, j \leq m, k \mid \binom{i}{j}$ 的 (i, j) 的数量，多组询问。
 $n, m \leq 2000, T \leq 10^4$ 。

求 $i \leq n, j \leq m, k \mid \binom{i}{j}$ 的 (i, j) 的数量，多组询问。
 $n, m \leq 2000, T \leq 10^4$ 。

求出模 k 意义下的杨辉三角，然后二维前缀和即可。

- n 个不同的物品排成一个环，旋转可得的算同一种方案，共有多少种方案？

- n 个不同的物品排成一个环，旋转可得的算同一种方案，共有多少种方案？

$$(n - 1)!$$

- n 个不同的物品排成一个环，旋转可得的算同一种方案，共有多少种方案？

$$(n-1)!$$

- n 个不同的物品选出 m 个排成一个环，旋转可得的算同一种方案，共有多少种方案？

- n 个不同的物品排成一个环，旋转可得的算同一种方案，共有多少种方案？

$$(n-1)!$$

- n 个不同的物品选出 m 个排成一个环，旋转可得的算同一种方案，共有多少种方案？

$$\binom{n}{m} (m-1)! = P_n^m / m$$

- n 种物品，每种物品都有无穷多个，选出 m 个物品排成一列，求方案数。

- n 种物品，每种物品都有无穷多个，选出 m 个物品排成一列，求方案数。

$$n^m$$

- n 种物品，每种物品都有无穷多个，选出 m 个物品排成一列，求方案数。

$$n^m$$

- 等价于求 $x_1 + x_2 + \cdots + x_n = m$ 非负整数解组数

- 等价于求 $x_1 + x_2 + \cdots + x_n = m$ 非负整数解组数

$$\binom{n+m-1}{n-1}$$

- 等价于求 $x_1 + x_2 + \cdots + x_n = m$ 非负整数解组数

$$\binom{n+m-1}{n-1}$$

- 如果每种物品都必须选择?

- 等价于求 $x_1 + x_2 + \cdots + x_n = m$ 非负整数解组数

$$\binom{n+m-1}{n-1}$$

- 如果每种物品都必须选择?
正整数解组数。

$$\binom{m-1}{n-1}$$

- n 种物品，第 i 种物品有 a_i 个，将所有物品排成一行，求方案数。

- n 种物品，第 i 种物品有 a_i 个，将所有物品排成一行，求方案数。

$$P\left(\sum_{i=1}^n a_i; a_1, a_2, \dots, a_n\right) = \frac{(\sum_{i=1}^n a_i)!}{\prod_{i=1}^n a_i!}$$

$$P(9; 2, 3, 4) = \frac{9!}{2!3!4!} = 1260.$$

- n 个球装入 n 个箱子，每个箱子装一个球。要求第 i 个球不能装入第 i 个箱子，求方案数。

- n 个球装入 n 个箱子，每个箱子装一个球。要求第 i 个球不能装入第 i 个箱子，求方案数。
- n 号球装入 i 号箱， i 号球装入 n 号箱： $D(n-2)$ 。

- n 个球装入 n 个箱子，每个箱子装一个球。要求第 i 个球不能装入第 i 个箱子，求方案数。
- n 号球装入 i 号箱， i 号球装入 n 号箱： $D(n-2)$ 。
- n 号球装入 i 号箱， i 号球装入 j 号箱：把 i 号箱和 n 号球“扔掉”， n 号箱当做 i 号箱（因为限制了 i 号球不能装入 n 号箱）， $D(n-1)$ 。

- n 个球装入 n 个箱子，每个箱子装一个球。要求第 i 个球不能装入第 i 个箱子，求方案数。
- n 号球装入 i 号箱， i 号球装入 n 号箱： $D(n-2)$ 。
- n 号球装入 i 号箱， i 号球装入 j 号箱：把 i 号箱和 n 号球“扔掉”， n 号箱当做 i 号箱（因为限制了 i 号球不能装入 n 号箱）， $D(n-1)$ 。

$$D(n) = (n-1)(D(n-1) + D(n-2)).$$

$$D(1) = 0, D(2) = 1, D(3) = 2, D(4) = 9, D(5) = 44, D(6) = 265 \dots$$

- n 个球装入 n 个箱子，每个箱子装一个球。要求第 i 个球不能装入第 i 个箱子，求方案数。
- n 号球装入 i 号箱， i 号球装入 n 号箱： $D(n-2)$ 。
- n 号球装入 i 号箱， i 号球装入 j 号箱：把 i 号箱和 n 号球“扔掉”， n 号箱当做 i 号箱（因为限制了 i 号球不能装入 n 号箱）， $D(n-1)$ 。

$$D(n) = (n-1)(D(n-1) + D(n-2)).$$

$$D(1) = 0, D(2) = 1, D(3) = 2, D(4) = 9, D(5) = 44, D(6) = 265 \dots$$

- n 个球， n 个箱子，每个箱子装一个球，另有 m 个多余的球。要求第 i 个球不能装入第 i 个箱子，求方案数。

- n 个球, n 个箱子, 每个箱子装一个球, 另有 m 个多余的球。要求第 i 个球不能装入第 i 个箱子, 求方案数。

$$D_m(n) = (n-1)(D_m(n-1) + D_m(n-2)) + mD_m(n-1).$$

$$D_m(n) = (n+m-1)D_m(n-1) + (n-1)D_m(n-2).$$

$$D_m(1) = m, D_m(2) = m^2 + m + 1$$

二项式定理

- 二项式就是形如 $(x + y)^n$ 的式子。它完全展开一共有 2^n 项。

二项式定理

- 二项式就是形如 $(x + y)^n$ 的式子。它完全展开一共有 2^n 项。
- 我们合并同类项时，考虑 $x^i y^{n-i}$ 有多少项。

二项式定理

- 二项式就是形如 $(x + y)^n$ 的式子。它完全展开一共有 2^n 项。
- 我们合并同类项时, 考虑 $x^i y^{n-i}$ 有多少项。
- 在 n 个数中选 i 个填 x , $n - i$ 个填 y , 因此有 $\binom{n}{i}$ 项。故

$$(x + y)^n = \sum_{i=0}^n \binom{n}{i} x^i y^{n-i}$$

二项式定理

- 二项式就是形如 $(x + y)^n$ 的式子。它完全展开一共有 2^n 项。
- 我们合并同类项时, 考虑 $x^i y^{n-i}$ 有多少项。
- 在 n 个数中选 i 个填 x , $n - i$ 个填 y , 因此有 $\binom{n}{i}$ 项。故

$$(x + y)^n = \sum_{i=0}^n \binom{n}{i} x^i y^{n-i}$$

- 例如 $(x + y)^5 = x^5 + 5x^4y + 10x^3y^2 + 10x^2y^3 + 5xy^4 + y^5$.
- 求 $(ax + by)^k$ 的 $x^n y^m$ 项系数?

二项式定理

- 二项式就是形如 $(x + y)^n$ 的式子。它完全展开一共有 2^n 项。
- 我们合并同类项时, 考虑 $x^i y^{n-i}$ 有多少项。
- 在 n 个数中选 i 个填 x , $n - i$ 个填 y , 因此有 $\binom{n}{i}$ 项。故

$$(x + y)^n = \sum_{i=0}^n \binom{n}{i} x^i y^{n-i}$$

- 例如 $(x + y)^5 = x^5 + 5x^4y + 10x^3y^2 + 10x^2y^3 + 5xy^4 + y^5$.
- 求 $(ax + by)^k$ 的 $x^n y^m$ 项系数?

$$\binom{k}{n} \binom{k}{m} a^n b^m$$

- 二项式定理的扩展。

$$(x_1 + x_2 + \cdots + x_m)^n = \sum_{\sum_{i=1}^m a_i = n} P(n; a_1, a_2, \cdots, a_m) x_1^{a_1} x_2^{a_2} \cdots x_n^{a_n}$$

例如 $(x + y + z)^3 =$
 $x^3 + y^3 + z^3 + 3x^2y + 3xy^2 + 3y^2z + 3yz^2 + 3z^2x + 3zx^2 + 6xyz.$

① 快速幂

② 组合数学
排列组合
容斥原理

③ 特殊的数

④ 质数，合数，约数，倍数

⑤ 同余方程

⑥ 欧拉函数

- 设 S_1, S_2 为有限集, 令 $|S|$ 表示 S 的大小, 则

$$|S_1 \cup S_2| = |S_1| + |S_2| - |S_1 \cap S_2|$$

- 设 S_1, S_2 为有限集, 令 $|S|$ 表示 S 的大小, 则

$$|S_1 \cup S_2| = |S_1| + |S_2| - |S_1 \cap S_2|$$

- 设 S_1, S_2, S_3 为有限集, 则

$$|S_1 \cup S_2 \cup S_3| = |S_1| + |S_2| + |S_3| - |S_1 \cap S_2| - |S_1 \cap S_3| - |S_2 \cap S_3| + |S_1 \cap S_2 \cap S_3|$$

- 设 S_1, S_2 为有限集, 令 $|S|$ 表示 S 的大小, 则

$$|S_1 \cup S_2| = |S_1| + |S_2| - |S_1 \cap S_2|$$

- 设 S_1, S_2, S_3 为有限集, 则

$$|S_1 \cup S_2 \cup S_3| = |S_1| + |S_2| + |S_3| - |S_1 \cap S_2| - |S_1 \cap S_3| - |S_2 \cap S_3| + |S_1 \cap S_2 \cap S_3|$$

- 一般地, 设 S_1, S_2, \dots, S_n 为 n 个有限集, 则

$$\begin{aligned} |\cup_{i=1}^n S_i| = & \sum_{1 \leq i \leq n} |S_i| - \sum_{1 \leq i < j \leq n} |S_i \cap S_j| + \sum_{1 \leq i < j < k \leq n} |S_i \cap S_j \cap S_k| \\ & - \dots + (-1)^{n-1} |\cap_{i=1}^n S_i| \end{aligned}$$

- 下面让我们来重新看一下错排问题。
- n 个人排队, 1 不能排在第一个, 2 不能排在第二个, $\dots\dots n$ 不能排在第 n 个,

错排问题的容斥解法

- 下面让我们来重新看一下错排问题。
- n 个人排队, 1 不能排在第一个, 2 不能排在第二个, $\dots\dots n$ 不能排在第 n 个,

$$D(n) = n! - \binom{n}{1}(n-1)! + \binom{n}{2}(n-2)! - \dots + (-1)^n \binom{n}{n} 0!$$

$$D(n) = n! \sum_{i=0}^n \frac{(-1)^i}{i!}$$

$$D_m(n) = P_{n+m}^n - \binom{n}{1} P_{n+m-1}^{n-1} + \binom{n}{2} P_{n+m-2}^{n-2} - \dots + (-1)^n \binom{n}{n} P_m^0$$

6 个人分成 A, B, C, D 四个小组，且每组都不能为空。求分组方案数。

例题-引入-第二类斯特林数

6 个人分成 A, B, C, D 四个小组，且每组都不能为空。求分组方案数。

$$4^6 - \binom{4}{1}3^6 + \binom{4}{2}2^6 - \binom{4}{3}1^6 = 4096 - 2916 + 384 - 4 = 1560.$$

如果四组相同？

例题-引入-第二类斯特林数

6 个人分成 A, B, C, D 四个小组，且每组都不能为空。求分组方案数。

$$4^6 - \binom{4}{1}3^6 + \binom{4}{2}2^6 - \binom{4}{3}1^6 = 4096 - 2916 + 384 - 4 = 1560.$$

如果四组相同？

$$\frac{1560}{4!} = 65.$$

n 个物品划分成 m 组，每组不为空且每组相同时的方案数称为“第二类斯特林数”，记作 $S(n, m)$ ，例如 $S(6, 4) = 65$ 。

第二类斯特林数

n 个物品划分成 m 组，每组不为空且每组相同时的方案数称为“第二类斯特林数”，记作 $S(n, m)$ ，例如 $S(6, 4) = 65$ 。

递推：考虑最后一个人是新开还是加入老组。

第二类斯特林数

n 个物品划分成 m 组，每组不为空且每组相同时的方案数称为“第二类斯特林数”，记作 $S(n, m)$ ，例如 $S(6, 4) = 65$ 。

递推：考虑最后一个人是新开还是加入老组。

$$S(n, m) = S(n - 1, m - 1) + mS(n - 1, m)$$

第二类斯特林数

n 个物品划分成 m 组，每组不为空且每组相同时的方案数称为“第二类斯特林数”，记作 $S(n, m)$ ，例如 $S(6, 4) = 65$ 。

递推：考虑最后一个人是新开还是加入老组。

$$S(n, m) = S(n-1, m-1) + mS(n-1, m)$$

容斥求通项：

$$S(n, m) = \frac{1}{m!} \sum_{i=0}^m (-1)^i \binom{m}{i} (m-i)^n$$

第二类斯特林数

n 个物品划分成 m 组，每组不为空且每组相同时的方案数称为“第二类斯特林数”，记作 $S(n, m)$ ，例如 $S(6, 4) = 65$ 。

递推：考虑最后一个人是新开还是加入老组。

$$S(n, m) = S(n-1, m-1) + mS(n-1, m)$$

容斥求通项：

$$S(n, m) = \frac{1}{m!} \sum_{i=0}^m (-1)^i \binom{m}{i} (m-i)^n$$

$O(n^2)$ 求出 $i \leq n, j \leq m$ 的所有 $S(i, j)$;

$O(n)$ 求出一项。

把正整数 n 划分为 m 个正整数的和，顺序不同的方案算同一种。

把正整数 n 划分为 m 个正整数的和，顺序不同的方案算同一种。

$$P(n, m) = P(n - 1, m - 1) + P(n - m, m), P(n, n) = 1, P(n, 1) = 1$$

$O(n^2)$ DP。

n 个球, m 个盒子, 求:

球相同/不相同, 盒子相同/不相同, 盒子可空/不可空的方案数。

$n, m \leq 2000$ 。

- 球不同，盒子不同，盒子可空：可重集排列问题；

盒子放球问题

- 球不同，盒子不同，盒子可空：可重集排列问题；
- 球相同，盒子不同，盒子不可空：可重集组合；

盒子放球问题

- 球不同，盒子不同，盒子可空：可重集排列问题；
- 球相同，盒子不同，盒子不可空：可重集组合；
- 球相同，盒子不同，盒子可空：可重集组合；

盒子放球问题

- 球不同，盒子不同，盒子可空：可重集排列问题；
- 球相同，盒子不同，盒子不可空：可重集组合；
- 球相同，盒子不同，盒子可空：可重集组合；
- 球相同，盒子相同，盒子可空：可重集组合；

盒子放球问题

- 球不同，盒子不同，盒子可空：可重集排列问题；
- 球相同，盒子不同，盒子不可空：可重集组合；
- 球相同，盒子不同，盒子可空：可重集组合；
- 球相同，盒子相同，盒子可空：可重集组合；
- 球相同，盒子相同，盒子不可空：划分数；

盒子放球问题

- 球不同，盒子不同，盒子可空：可重集排列问题；
- 球相同，盒子不同，盒子不可空：可重集组合；
- 球相同，盒子不同，盒子可空：可重集组合；
- 球相同，盒子相同，盒子可空：可重集组合；
- 球相同，盒子相同，盒子不可空：划分数；
- 球不同，盒子相同，盒子不可空：第二类斯特林数；

盒子放球问题

- 球不同，盒子不同，盒子可空：可重集排列问题；
- 球相同，盒子不同，盒子不可空：可重集组合；
- 球相同，盒子不同，盒子可空：可重集组合；
- 球相同，盒子相同，盒子可空：可重集组合；
- 球相同，盒子相同，盒子不可空：划分数；
- 球不同，盒子相同，盒子不可空：第二类斯特林数；
- 球不同，盒子不同，盒子不可空：第二类斯特林数；

盒子放球问题

- 球不同，盒子不同，盒子可空：可重集排列问题；
- 球相同，盒子不同，盒子不可空：可重集组合；
- 球相同，盒子不同，盒子可空：可重集组合；
- 球相同，盒子相同，盒子可空：可重集组合；
- 球相同，盒子相同，盒子不可空：划分数；
- 球不同，盒子相同，盒子不可空：第二类斯特林数；
- 球不同，盒子不同，盒子不可空：第二类斯特林数；
- 球不同，盒子相同，盒子可空：第二类斯特林数

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}$$

$$\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$$

$$\sum_{i=1}^n i^3 = \frac{n^2(n+1)^2}{4}$$

但是 $k \geq 4$ 时, 我们就没有特别简便的公式了。

$$\begin{aligned} n^{k+1} &= \sum_{i=1}^n i^{k+1} - (i-1)^{k+1} \\ &= \sum_{i=1}^n \left(i^{k+1} - \sum_{j=0}^{k+1} (-1)^{k+1-j} \binom{k+1}{j} i^j \right) \\ &= \sum_{i=1}^n \sum_{j=0}^k (-1)^{k-j} \binom{k+1}{j} i^j \\ &= \sum_{j=0}^k (-1)^{k-j} \binom{k+1}{j} \sum_{i=1}^n i^j = n^{k+1} \end{aligned}$$

设 $S_k = \sum_{i=1}^n i^k$, $S_k = \frac{1}{k+1} (n^{k+1} - \sum_{j=0}^{k-1} (-1)^{k-j} \binom{k+1}{j} S_j)$

初始: $S_0 = n$

$$\binom{n}{m} \bmod p = \binom{n/p}{m/p} \binom{n \% p}{m \% p} \bmod p$$

```
1 ll C(int n,int m){
2     if(n<m) return 0;
3     return fac[n]*inf[m]%p*inf[n-m]%p;
4 }
5 ll lucas(int n,int m){
6     if(n<m) return 0;
7     if(n<p) return C(n,m);
8     return lucas(n/p,m/p)*C(n%p,m%p)%p;
9 }
```

Lucas 定理证明

设 $n = sp + q$, $m = tp + r$, 且 $0 \leq q, r < p$, 构造多项式

$$(1+x)^n = (1+x)^{sp+q} = ((1+x)^p)^s (1+x)^q$$

因为对任意的 $1 < i < p$, $p \mid \binom{p}{i}$, 所以

$$(1+x)^p = 1 + x^p \pmod{p}$$

所以

$$(1+x)^{sp+q} = (1+x^p)^s (1+x)^q = \sum_{i=0}^s \binom{s}{i} x^{ip} \cdot \sum_{j=0}^q \binom{q}{j} x^j \pmod{p}$$

比较等式两端 x^{tp+r} 的系数, 则有

$$\binom{sp+q}{tp+r} = \binom{s}{t} \binom{q}{r}$$

即定理得证。

组合数前缀和 - [SHOI2015] 超能粒子炮 · 改

求 $\sum_{i=0}^k \binom{n}{i} \bmod p$, $n, k \leq 10^9, p = 2333$, T 组询问, $T \leq 10^5$ 。

Solution:

每 p 个分一段:

$$\sum_{i=0}^k \binom{n/p}{i/p} \binom{n \% p}{i \% p}$$

$$\sum_{i=0}^{k/p-1} \binom{n/p}{i} \sum_{j=0}^{p-1} \binom{n \% p}{j} + \binom{n/p}{k/p} \sum_{j=0}^{k \% p} \binom{n \% p}{j}$$

$$\text{设 } f(n, k) = \sum_{i=0}^k \binom{n}{i}$$

$$f(n, k) = f(n/p, k/p - 1) f(n \% p, p - 1) + \binom{n/p}{k/p} f(n \% p, k \% p)$$

$$O(p^2 + T \log n)$$

① 快速幂

② 组合数学

③ 特殊的数

斐波那契数

卡特兰数

④ 质数，合数，约数，倍数

⑤ 同余方程

⑥ 欧拉函数

① 快速幂

② 组合数学

③ 特殊的数
斐波那契数
卡特兰数

④ 质数，合数，约数，倍数

⑤ 同余方程

⑥ 欧拉函数

$$F_0 = 0, F_1 = 1, F_n = F_{n-1} + F_{n-2} (n \geq 3, n \in \mathcal{N})$$

$$F_0 = 0, F_1 = 1, F_n = F_{n-1} + F_{n-2} (n \geq 3, n \in \mathcal{N})$$

通项公式:

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

- $F_1 + F_2 + \cdots + F_n = F_{n+2} - 1$

- $F_1 + F_2 + \cdots + F_n = F_{n+2} - 1$
- $F_1^2 + F_2^2 + \cdots + F_n^2 = F_n F_{n+1}$

- $F_1 + F_2 + \cdots + F_n = F_{n+2} - 1$
- $F_1^2 + F_2^2 + \cdots + F_n^2 = F_n F_{n+1}$
- $F_1 + F_3 + F_5 + \cdots + F_{2n-1} = F_{2n}$

- $F_1 + F_2 + \cdots + F_n = F_{n+2} - 1$
- $F_1^2 + F_2^2 + \cdots + F_n^2 = F_n F_{n+1}$
- $F_1 + F_3 + F_5 + \cdots + F_{2n-1} = F_{2n}$
- $F_2 + F_4 + F_6 + \cdots + F_{2n} = F_{2n+1} - 1$

- $F_1 + F_2 + \cdots + F_n = F_{n+2} - 1$
- $F_1^2 + F_2^2 + \cdots + F_n^2 = F_n F_{n+1}$
- $F_1 + F_3 + F_5 + \cdots + F_{2n-1} = F_{2n}$
- $F_2 + F_4 + F_6 + \cdots + F_{2n} = F_{2n+1} - 1$
- $F_n = F_m F_{n-m+1} + F_{m-1} F_{n-m}$

- $F_1 + F_2 + \cdots + F_n = F_{n+2} - 1$
- $F_1^2 + F_2^2 + \cdots + F_n^2 = F_n F_{n+1}$
- $F_1 + F_3 + F_5 + \cdots + F_{2n-1} = F_{2n}$
- $F_2 + F_4 + F_6 + \cdots + F_{2n} = F_{2n+1} - 1$
- $F_n = F_m F_{n-m+1} + F_{m-1} F_{n-m}$
- $F_{n-1} F_{n+1} = F_n^2 + (-1)^n$

- 相邻项互质: $\gcd(F_n, F_{n-1}) = 1$;

- 相邻项互质: $\gcd(F_n, F_{n-1}) = 1$;
- 辗转相除法: $\gcd(F_n, F_m) = F_{\gcd(n,m)}$;

- 相邻项互质: $\gcd(F_n, F_{n-1}) = 1$;
- 辗转相除法: $\gcd(F_n, F_m) = F_{\gcd(n,m)}$;
- 整除的等价表示: $n|m \Leftrightarrow F_n|F_m$.

① 快速幂

② 组合数学

③ 特殊的数

斐波那契数

卡特兰数

④ 质数，合数，约数，倍数

⑤ 同余方程

⑥ 欧拉函数

卡特兰数 $C(n)$ 定义为凸 $n + 2$ 边形不同的三角剖分数。

$$C(1) = 1, C(2) = 2, C(3) = 5, C(4) = 14, C(5) = 42, C(6) = 132, \dots$$

卡特兰数 $C(n)$ 定义为凸 $n + 2$ 边形不同的三角剖分数。

$$C(1) = 1, C(2) = 2, C(3) = 5, C(4) = 14, C(5) = 42, C(6) = 132, \dots$$

根据这个定义，可得递推公式：

$$C(n) = \sum_{i=0}^{n-1} C(i)C(n-i-1)$$

卡特兰数有若干种等价定义。例如：

- n 个数按 $1 \sim n$ 的次序入栈，不同的出栈序列数为 $C(n)$ 。

卡特兰数有若干种等价定义。例如：

- n 个数按 $1 \sim n$ 的次序入栈，不同的出栈序列数为 $C(n)$ 。
- n 个点的二叉搜索树个数（即根的序号大于左儿子，小于右儿子）为 $C(n)$ 。

卡特兰数有若干种等价定义。例如：

- n 个数按 $1 \sim n$ 的次序入栈，不同的出栈序列数为 $C(n)$ 。
- n 个点的二叉搜索树个数（即根的序号大于左儿子，小于右儿子）为 $C(n)$ 。
- n 个 1 和 n 个 -1 组成一个长 $2n$ 的序列，且满足序列的所有前缀和都非负，序列数量为 $C(n)$ 。

$$C(n) = \binom{2n}{n} - \binom{2n}{n+1}$$

$$C(n) = \frac{\binom{2n}{n}}{n+1}$$

$$C(n) = C(n-1) \cdot \frac{4n-2}{n+1}.$$

- ① 快速幂
- ② 组合数学
- ③ 特殊的数
- ④ 质数，合数，约数，倍数
筛法
- ⑤ 同余方程
- ⑥ 欧拉函数

若存在 d 使得 $y = dx$, 则称 x 整除 y , 记作 $x|y$ 。

若存在 d 使得 $y = dx$ ，则称 x 整除 y ，记作 $x|y$ 。

若 $x|y$ ，则称 x 是 y 的约数（又称因数）， y 是 x 的倍数。

若存在 d 使得 $y = dx$ ，则称 x 整除 y ，记作 $x|y$ 。

若 $x|y$ ，则称 x 是 y 的约数（又称因数）， y 是 x 的倍数。

若 $p > 1$ 且只有 1 和自身两个约数，则称 p 是质数。

若存在 d 使得 $y = dx$ ，则称 x 整除 y ，记作 $x|y$ 。

若 $x|y$ ，则称 x 是 y 的约数（又称因数）， y 是 x 的倍数。

若 $p > 1$ 且只有 1 和自身两个约数，则称 p 是质数。

若 $x > 1$ 且有超过两个约数，则称 x 是合数。

若存在 d 使得 $y = dx$ ，则称 x 整除 y ，记作 $x|y$ 。

若 $x|y$ ，则称 x 是 y 的约数（又称因数）， y 是 x 的倍数。

若 $p > 1$ 且只有 1 和自身两个约数，则称 p 是质数。

若 $x > 1$ 且有超过两个约数，则称 x 是合数。

0, 1 不是质数也不是合数。

任意一个数 x 均可分解为若干质数幂次的乘积。即

$$x = \prod_{i=1}^m p_i^{c_i}$$

其中 p_1, \dots, p_m 是不同的质数, $c_1, \dots, c_m \geq 1$ 。

$O(\sqrt{n})$ 判定法，每个数最多只有一个 $> \sqrt{n}$ 的因数。

单个质数判定

$O(\sqrt{n})$ 判定法，每个数最多只有一个 $> \sqrt{n}$ 的因数。

```
1 bool prime(int n){  
2     for(int i=2;i*i<=n;++i) if(!(n%i)) return 1;  
3     return 0;  
4 }
```

单个质数判定

$O(\sqrt{n})$ 判定法, 每个数最多只有一个 $> \sqrt{n}$ 的因数。

```
1 bool prime(int n){  
2     for(int i=2;i*i<=n;++i) if(!(n%i)) return 1;  
3     return 0;  
4 }
```

上述过程用于分解质因数:

```
1 for(int i=2;i*i<=n;++i) if(!(n%i)){  
2     while(!(n%i)) n/=i, ++cnt;  
3     //do something for (i,cnt)  
4 }  
5 if(n>1) //do something for (n,1)
```

单个质数判定

$O(\sqrt{n})$ 判定法, 每个数最多只有一个 $> \sqrt{n}$ 的因数。

```
1 bool prime(int n){  
2     for(int i=2;i*i<=n;++i) if(!(n%i)) return 1;  
3     return 0;  
4 }
```

上述过程用于分解质因数:

```
1 for(int i=2;i*i<=n;++i) if(!(n%i)){  
2     while(!(n%i)) n/=i, ++cnt;  
3     //do something for (i,cnt)  
4 }  
5 if(n>1) //do something for (n,1)
```

更高效的质数判定: Miller-Rabin 二次探测法, 复杂度 $O(\log^2 n)$;

单个质数判定

$O(\sqrt{n})$ 判定法, 每个数最多只有一个 $> \sqrt{n}$ 的因数。

```
1 bool prime(int n){  
2     for(int i=2; i*i<=n; ++i) if(!(n%i)) return 1;  
3     return 0;  
4 }
```

上述过程用于分解质因数:

```
1 for(int i=2; i*i<=n; ++i) if(!(n%i)){  
2     while(!(n%i)) n/=i, ++cnt;  
3     //do something for (i, cnt)  
4 }  
5 if(n>1) //do something for (n, 1)
```

更高效的质数判定: Miller-Rabin 二次探测法, 复杂度 $O(\log^2 n)$;

更高效的质因数分解: Pollard-Rho 算法, 复杂度 $O(n^{\frac{1}{4}})$ 。

- ① 快速幂
- ② 组合数学
- ③ 特殊的数
- ④ 质数，合数，约数，倍数
筛法
- ⑤ 同余方程
- ⑥ 欧拉函数

1- n 中所有质数的判定

如何快速找到 1 到 n 中的全部质数? $n \leq 10^7$ 。

1- n 中所有质数的判定

如何快速找到 1 到 n 中的全部质数? $n \leq 10^7$ 。
如果对于每个数都 $O(\sqrt{n})$ 暴力判定, 复杂度 $O(n\sqrt{n})$ 无法接受。

先把所有数列出来，然后
划掉除 2 外所有被 2 整除的数字；
划掉除 3 外所有被 3 整除的数字；
划掉除 5 外所有被 5 整除的数字（为什么不是 4？）；
... ..
划到 \sqrt{n} 为止，剩下的就都是质数了。
复杂度 $O(n \log \log n)$ 。

埃氏筛的思路是先枚举质数 p ，再枚举另一个约数 i ，将所有 $i \times p$ 都标记为合数。

欧拉筛则相反，先枚举另一个约数 i ，再枚举质数 p 。只不过我们强制这个 p 是 $i \times p$ 的最小质因数。

当 p 枚举到 i 的质因数时， p 就不再是 $i \times p$ 的最小质因数了（可以退出了）。因为每个数只会被筛一次，所以复杂度为 $O(n)$ 。

欧拉筛

埃氏筛的思路是先枚举质数 p ，再枚举另一个约数 i ，将所有 $i \times p$ 都标记为合数。

欧拉筛则相反，先枚举另一个约数 i ，再枚举质数 p 。只不过我们强制这个 p 是 $i \times p$ 的最小质因数。

当 p 枚举到 i 的质因数时， p 就不再是 $i \times p$ 的最小质因数了（可以退出了）。因为每个数只会被筛一次，所以复杂度为 $O(n)$ 。

```
1 bool np[N];
2 int pri[N], cnt=0;
3 void sieve(int n){
4     for(int i=2; i<=n; ++i){
5         if(!np[i]) pri[++cnt]=i;
6         for(int j=1; j<=cnt; ++j){
7             np[i*pri[j]]=1;
8             if(!(i%pri[j])) break;
9         }
10    }
11 }
```

求区间 $[L, R]$ 之间的素数个数, $L, R \leq 10^{12}, R - L \leq 10^6$ 。

求区间 $[L, R]$ 之间的素数个数, $L, R \leq 10^{12}, R - L \leq 10^6$ 。

10^{12} 的范围不允许我们从头筛, 10^6 的区间长度不允许我们判断每个质数是否为质数。

那我们就只能 “从中间筛”!

我们用小于 \sqrt{R} 的所有质数去筛 L 到 R 的数, 如果整除则标记为合数。
复杂度为 $O(n \log n)$ ($n = 10^6 = \sqrt{10^{12}}$)

给 n 个数，对每个 i ，求满足 $a_j | a_i, j \neq i$ 的 j 的数量（即有多少个数是它的约数）。

$$1 \leq n \leq 10^6, 1 \leq a_i \leq 10^6$$

因为 $a_i \leq 10^6$ ，所以考虑把每个数的因数都求出来，并预处理每个数的个数 (*count* 数组)

复杂度降为 $O(n\sqrt{M})$ ，然而还是过不去（众所周知除法和取模比加减乘要慢三倍， 10^9 次除法或取模就别想了）。

因为 $a_i \leq 10^6$ ，所以考虑把每个数的因数都求出来，并预处理每个数的个数 (*count* 数组)

复杂度降为 $O(n\sqrt{M})$ ，然而还是过不去（众所周知除法和取模比加减乘要慢三倍， 10^9 次除法或取模就别想了）。

枚举倍数：

我们把枚举因数的两层循环“倒过来”，就变成了枚举倍数——

```
1 for (int i=1; i<=M; ++i)
2   for (int j=i; j<=M; j+=i)
3     //do something
```

因为 $a_i \leq 10^6$ ，所以考虑把每个数的因数都求出来，并预处理每个数的个数 (*count* 数组)

复杂度降为 $O(n\sqrt{M})$ ，然而还是过不去（众所周知除法和取模比加减乘要慢三倍， 10^9 次除法或取模就别想了）。

枚举倍数：

我们把枚举因数的两层循环“倒过来”，就变成了枚举倍数——

```
1 for (int i=1; i<=M; ++i)
2   for (int j=i; j<=M; j+=i)
3     //do something
```

这样就可以得到 $1 \cdots 10^6$ 每个数在 $\{a_i\}$ 里的约数个数。

因为 $a_i \leq 10^6$ ，所以考虑把每个数的因数都求出来，并预处理每个数的个数 (*count* 数组)

复杂度降为 $O(n\sqrt{M})$ ，然而还是过不去（众所周知除法和取模比加减乘要慢三倍， 10^9 次除法或取模就别想了）。

枚举倍数：

我们把枚举因数的两层循环“倒过来”，就变成了枚举倍数——

```
1 for(int i=1;i<=M;++i)
2 for(int j=i;j<=M;j+=i)
3     //do something
```

这样就可以得到 $1 \cdots 10^6$ 每个数在 $\{a_i\}$ 里的约数个数。

我们来分析一下这样做的复杂度。内层循环一共进行了

$$M + \frac{M}{2} + \frac{M}{3} + \cdots + \frac{M}{M} = M(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{M}) = O(M \log M) \text{ 次!}$$

于是我们的复杂度成功降到了 $O(n + M \log M)$ 。

若 $d|x, d|y$ 同时成立, 则称 d 是 x, y 的公约数。最大公约数记为 $\gcd(a, b)$ 。

若 $d|x, d|y$ 同时成立, 则称 d 是 x, y 的公约数。最大公约数记为 $\gcd(a, b)$ 。

若 $x|m, y|m$ 同时成立, 则称 m 是 x, y 的公倍数。最小公倍数记为 $\text{lcm}(a, b)$ 。

若 $d|x, d|y$ 同时成立, 则称 d 是 x, y 的公约数。最大公约数记为 $\gcd(a, b)$ 。

若 $x|m, y|m$ 同时成立, 则称 m 是 x, y 的公倍数。最小公倍数记为 $\text{lcm}(a, b)$ 。

辗转相除求最大公约数

```
int gcd(int x, int y){return y?gcd(y, x%y):x;}
```

我们一般记

$$\sigma_m(n) = \prod_{d|n} d^m$$

约数个数 $\sigma_0(n)$ 和约数和 $\sigma_1(n)$

我们一般记

$$\sigma_m(n) = \prod_{d|n} d^m$$

常用的： $\sigma_0(n)$ 表示约数个数， $\sigma_1(n)$ 表示约数和。

约数个数 $\sigma_0(n)$ 和约数和 $\sigma_1(n)$

我们一般记

$$\sigma_m(n) = \prod_{d|n} d^m$$

常用的： $\sigma_0(n)$ 表示约数个数， $\sigma_1(n)$ 表示约数和。

若 $n = \prod_{i=1}^k p_i^{c_i}$ ，则考虑 n 的约数的唯一分解，每个 p_i 的幂次都 $\leq c_i$ ，因此由乘法原理可得

$$\sigma_0(n) = \prod_{i=1}^k (c_i + 1)$$

$$\sigma_1(n) = \prod_{i=1}^k (1 + p_i + p_i^2 + \cdots + p_i^{c_i}) = \prod_{i=1}^k \frac{p_i^{c_i+1} - 1}{p_i - 1}$$

约数个数 $\sigma_0(n)$ 和约数和 $\sigma_1(n)$

我们一般记

$$\sigma_m(n) = \prod_{d|n} d^m$$

常用的： $\sigma_0(n)$ 表示约数个数， $\sigma_1(n)$ 表示约数和。

若 $n = \prod_{i=1}^k p_i^{c_i}$ ，则考虑 n 的约数的唯一分解，每个 p_i 的幂次都 $\leq c_i$ ，因此由乘法原理可得

$$\sigma_0(n) = \prod_{i=1}^k (c_i + 1)$$

$$\sigma_1(n) = \prod_{i=1}^k (1 + p_i + p_i^2 + \cdots + p_i^{c_i}) = \prod_{i=1}^k \frac{p_i^{c_i+1} - 1}{p_i - 1}$$

因此我们对 n 分解质因数后，即可直接得到 n 的约数个数、约数和。

约数个数 $\sigma_0(n)$ 和约数和 $\sigma_1(n)$

我们一般记

$$\sigma_m(n) = \prod_{d|n} d^m$$

常用的： $\sigma_0(n)$ 表示约数个数， $\sigma_1(n)$ 表示约数和。

若 $n = \prod_{i=1}^k p_i^{c_i}$ ，则考虑 n 的约数的唯一分解，每个 p_i 的幂次都 $\leq c_i$ ，因此由乘法原理可得

$$\sigma_0(n) = \prod_{i=1}^k (c_i + 1)$$

$$\sigma_1(n) = \prod_{i=1}^k (1 + p_i + p_i^2 + \cdots + p_i^{c_i}) = \prod_{i=1}^k \frac{p_i^{c_i+1} - 1}{p_i - 1}$$

因此我们对 n 分解质因数后，即可直接得到 n 的约数个数、约数和。
当 n 很大，而 n 的质因数分解已知时，上述两个公式可以快速求得 n 的约数个数、约数和（举个最简单的例子，求 $n!$ 的约数和， $n < 10^6$ ）。

已知 $\sigma_1(x) = n$, 求所有的 x 。 T 组询问。
 $n \leq 2 \times 10^9$

将 n 分解 (不一定是质因数), 然后考虑把 n 写成约数和公式的形式。
例如:

$$n = 42 = 6 \times 7 = (1 + 5) \times (1 + 2 + 2^2) \rightarrow x = 5 \times 2^2 = 20$$

$$n = 42 = 3 \times 14 = (1 + 2) \times (1 + 13) \rightarrow x = 2 \times 13 = 26$$

$$n = 42 = 1 + 41 \rightarrow x = 41$$

因此我们用 DFS 分解 n 。从 1 到 \sqrt{n} 枚举 x 的质因数及其幂次, 并检验 $1 + p + \dots + p^c$ 是否可整除 n , 若能整除则搜下一层。

如果 $n = 1$ 则将 x 加入答案并返回; 如果 $n - 1$ 是质数则直接给当前的 x 乘 $n - 1$ 并加入答案。

① 快速幂

② 组合数学

③ 特殊的数

④ 质数，合数，约数，倍数

⑤ 同余方程

线性同余方程

线性同余方程组

⑥ 欧拉函数

同余方程包括：线性同余数方程（`exgcd` 算法），线性同余方程组（中国剩余定理 CRT，扩展中国剩余定理），指数同余方程（BSGS 算法），高次同余方程（阶和原根）。

该部分涉及的数学知识较多，时间关系我们今天只介绍最简单的 `exgcd` 算法。CRT 算法作为了解。

① 快速幂

② 组合数学

③ 特殊的数

④ 质数，合数，约数，倍数

⑤ 同余方程

线性同余方程

线性同余方程组

⑥ 欧拉函数

若 $(a, b) = d$ ，则对任意整数 x, y ，均有 $d \mid (ax + by)$ ，且二元一次不定方程 $ax + by = d$ 必有解。特别地， $(a, b) = 1$ 当且仅当 $ax + by = 1$ 有解。

关于 x 的形如 $ax = b(\text{mod } m)$ 的方程称为线性同余方程。

关于 x 的形如 $ax = b(\text{mod } m)$ 的方程称为线性同余方程。
由乘法逆元的定义可知，
若 $(a, m) = 1$ ，则该同余方程的解为 $x = ba^{-1} \text{ mod } m$ 。

关于 x 的形如 $ax = b \pmod{m}$ 的方程称为线性同余方程。

由乘法逆元的定义可知，

若 $(a, m) = 1$ ，则该同余方程的解为 $x = ba^{-1} \pmod{m}$ 。

若 $(a, m) \neq 1$ ，设 $d = (a, m)$ 。若 $d \mid b$ ，则方程两边及模数同时除 d 即可；否则方程无解。

例如 $3x = 2 \pmod{6}$ 就是无解的。

形如 $ax + by = c$ 的方程称为二元一次不定方程。

因为 $ax + by = c$ 可写作 $ax = c \pmod{b}$,

因此对于二元一次不定方程 $ax + by = c$, 方程两边同时除以 (a, b, c) 后, 即可得到其通解为 $x = x_0 + bt, y = y_0 - at$, 其中 $x_0 = ca^{-1}$

$\pmod{b}, y_0 = \frac{ax_0 - c}{b} \pmod{a}$ 。

扩展欧几里得算法

之前提到过，当 m 很大的时候，通过求 m 的欧拉函数得到 a 的逆元复杂度过高。

扩展欧几里得 (exgcd) 算法就是通过辗转相除得到方程 $ax + by = 1$ ($(a, b) = 1$) 的一组解，从而得到 a 模 b 意义下的逆元，解出线性同余方程或二元一次不定方程。

扩展欧几里得算法

之前提到过，当 m 很大的时候，通过求 m 的欧拉函数得到 a 的逆元复杂度过高。

扩展欧几里得 (exgcd) 算法就是通过辗转相除得到方程 $ax + by = 1$ ($(a, b) = 1$) 的一组解，从而得到 a 模 b 意义下的逆元，解出线性同余方程或二元一次不定方程。

我们现有方程

$$ax + by = 1$$

扩展欧几里得算法

之前提到过，当 m 很大的时候，通过求 m 的欧拉函数得到 a 的逆元复杂度过高。

扩展欧几里得 (exgcd) 算法就是通过辗转相除得到方程 $ax + by = 1$ ($(a, b) = 1$) 的一组解，从而得到 a 模 b 意义下的逆元，解出线性同余方程或二元一次不定方程。

我们现有方程

$$ax + by = 1$$

辗转相除时，我们令

$$a_1 = b, b_1 = a \pmod{b}$$

即

$$a_1 = b, b_1 = a - kb$$

其中 $k = \lfloor \frac{a}{b} \rfloor$,

则若 $a_1x + b_1y = 1$ 的解为 $x = x_1, y = y_1$, 即

$$bx_1 + (a - kb)y_1 = 1$$

即

$$ay_1 + b(x_1 - ky_1) = 1$$

则

$$x = y_1, y = x_1 - ky_1$$

当除到最后一步时, 有 $a = 1, b = 0$, 此时易知 $x = 1, y = 0$ 是一组解。
因此直接返回即可。

```
1 int exgcd(int a, int b, int& x, int& y){  
2     if(!b){x=1, y=0; return;}  
3     exgcd(b, a%b, y, x);  
4     y-=x*(a/b);  
5 }
```

```
1 int exgcd(int a,int b,int& x,int& y){  
2     if(!b){x=1,y=0;return;}  
3     exgcd(b,a%b,y,x);  
4     y-=x*(a/b);  
5 }
```

用 exgcd 求逆元:

$$a(a^{-1}) + xM = 1$$

```
1 int exgcd(int a, int b, int& x, int& y){  
2     if(!b){x=1, y=0; return;}  
3     exgcd(b, a%b, y, x);  
4     y-=x*(a/b);  
5 }
```

用 exgcd 求逆元:

$$a(a^{-1}) + xM = 1$$

其中 M 是模数, a 是要求逆元的数, 求解上述二元一次不定方程, 得到 a^{-1} 和 x 的值。

① 快速幂

② 组合数学

③ 特殊的数

④ 质数，合数，约数，倍数

⑤ 同余方程

线性同余方程

线性同余方程组

⑥ 欧拉函数

形如

$$\begin{cases} x \equiv a_1 & (\text{mod } m_1) \\ x \equiv a_2 & (\text{mod } m_2) \\ \dots \\ x \equiv a_n & (\text{mod } m_n) \end{cases}$$

的方程组称为线性同余方程组。

形如

$$\begin{cases} x \equiv a_1 & (\text{mod } m_1) \\ x \equiv a_2 & (\text{mod } m_2) \\ \dots \\ x \equiv a_n & (\text{mod } m_n) \end{cases}$$

的方程组称为线性同余方程组。

该方程组的解形如 $x = A \pmod{M}$, 其中 $M = \text{lcm}(m_1, m_2, \dots, m_n)$ 。

中国剩余定理 (CRT)

当 m_1, m_2, \dots, m_n 两两互质时, 我们可以直接用公式求解。

设 $M = m_1 m_2 \cdots m_n = \text{lcm}(m_1, m_2, \dots, m_n)$, $M_i = \frac{M}{m_i}$, $N_i = M_i^{-1} \pmod{m_i}$, 则

$$x = \sum_{i=1}^n a_i M_i N_i \pmod{M}$$

① 快速幂

② 组合数学

③ 特殊的数

④ 质数，合数，约数，倍数

⑤ 同余方程

⑥ 欧拉函数

积性函数，线性筛

欧拉定理/费马小定理

定义 n 的欧拉函数为 $1 \sim n$ 中与 n 互质的数的数量, 记作 $\varphi(n)$ 。
设 $n = \prod_{i=1}^k p_i^{c_i}$, 则

$$\varphi(n) = n \prod_{i=1}^k \left(1 - \frac{1}{p_i}\right) = \prod_{i=1}^k p_i^{c_i-1} (p_i - 1)$$

根据上面的公式, 我们可以 $O(\sqrt{n})$ 求出 $\varphi(n)$ 。

- ① 快速幂
- ② 组合数学
- ③ 特殊的数
- ④ 质数，合数，约数，倍数
- ⑤ 同余方程
- ⑥ 欧拉函数
 - 积性函数，线性筛
 - 欧拉定理/费马小定理

如果一个函数 f 满足以下性质：

- $f(1) = 1$
- 若 $\gcd(p, q) = 1$ ，则 $f(pq) = f(p)f(q)$

则称 f 是积性函数。

如果一个函数 f 满足以下性质：

- $f(1) = 1$
- 若 $\gcd(p, q) = 1$ ，则 $f(pq) = f(p)f(q)$

则称 f 是积性函数。

例如 $1, Id, \varphi, \sigma_0, \sigma_1$ 都是积性函数。

只要 $f(p^k)$ 可以 $O(1)$ 算出, 积性函数就可以使用欧拉筛在线性时间得到 $1 \sim n$ 的结果。

```
1 f[1]=1;
2 for(int i=2;i<=n;++i){
3     if(!np[i])f[i]=F(p,1),c[i]=1;
4     for(int j=1;j<=cnt&& i*pri[j]<=n;++j){
5         int p=pri[j],ip=i*p;
6         np[ip]=1;
7         if(!(i%p)){
8             f[ip]=f[i]/F(p,c[i])*F(p,c[i]+1);
9             c[ip]=c[i]+1;
10            break;
11        }
12        f[ip]=f[i]*f[p];
13        c[ip]=1;
14    }
15 }
```

其中 $f(i)$ 是积性函数值, $F(p,k)$ 是 $f(p^k)$ 的值 (可以直接算), $c(i)$ 是 i 最小质因数的幂次。

线性筛简化版本

如果 $\frac{f(p^{k+1})}{f(p^k)}$ 为定值 $F(p)$, 则无需存 c 。

```
1 f[1]=1;
2 for(int i=2;i<=n;++i){
3     if(!np[i])f[i]=P(p),c[i]=1;
4     for(int j=1;j<=cnt&& i*pri[j]<=n;++j){
5         int p=pri[j],ip=i*p;
6         np[ip]=1;
7         if(!(i%p)){
8             f[ip]=f[i]*F(p);
9             break;
10        }
11        f[ip]=f[i]*f[p];
12    }
13 }
```

其中 $F1(p)$ 是 $f(p)$ 的值。

下面我们通过 $\varphi, \sigma_0, \sigma_1, \mu$ (莫比乌斯函数) 举例说明:

下面我们通过 $\varphi, \sigma_0, \sigma_1, \mu$ (莫比乌斯函数) 举例说明:

$$\varphi : F1(p) = p - 1, F(p) = p$$

下面我们通过 $\varphi, \sigma_0, \sigma_1, \mu$ (莫比乌斯函数) 举例说明:

$$\varphi : F1(p) = p - 1, F(p) = p$$

$$\sigma_0 : F(p, k) = k + 1$$

下面我们通过 $\varphi, \sigma_0, \sigma_1, \mu$ (莫比乌斯函数) 举例说明:

$$\varphi : F1(p) = p - 1, F(p) = p$$

$$\sigma_0 : F(p, k) = k + 1$$

$$\sigma_1 : F(p, k) = 1 + p + \cdots + p^k$$

下面我们通过 $\varphi, \sigma_0, \sigma_1, \mu$ (莫比乌斯函数) 举例说明:

$$\varphi : F1(p) = p - 1, F(p) = p$$

$$\sigma_0 : F(p, k) = k + 1$$

$$\sigma_1 : F(p, k) = 1 + p + \cdots + p^k$$

$\mu(n)$ 的定义:

$$\mu(n) = \begin{cases} 1 & n = 1 \\ (-1)^n & n = p_1 \dots p_k \\ 0 & \text{others} \end{cases}$$

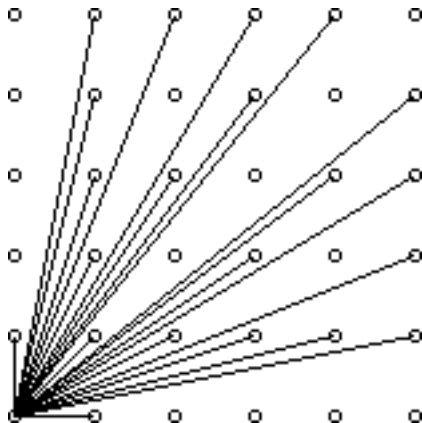
即有平方因子的数的莫比乌斯函数值均为 0。

$$\mu : F1(p) = -1, F(p) = 0$$

[SDOI2008] 仪仗队

作为体育委员，C 君负责这次运动会仪仗队的训练。仪仗队是由学生组成的 $N \times N$ 的方阵，为了保证队伍在行进中整齐划一，C 君会跟在仪仗队的左后方，根据其视线所及的学生人数来判断队伍是否整齐（如下图）。

$n \leq 10^6$ 。



我们不妨把 $(1, 1)$ 看做原点，所有点的坐标就是 $(0, 0)$ 到 $(n-1, n-1)$ ， (i, j) 可见当且仅当 $(i, j) = 1$ 。

因此我们要求的就是（为方便起见令 $n \leftarrow n-1$ ）

$$\sum_{i=1}^n \sum_{j=1}^n [\gcd(i, j) == 1] + 2$$

\gcd 矩阵对角线两边完全对称，因此只需计算

$$2 \sum_{i=1}^n \sum_{j=1}^i [\gcd(i, j) == 1] + 1$$

注意对角线上的 $(1, 1)$ 被重复计算，所以要减 1。
而根据 $\varphi(i)$ 的定义有

$$\sum_{j=1}^i [\gcd(i, j) == 1] = \varphi(i)$$

- ① 快速幂
- ② 组合数学
- ③ 特殊的数
- ④ 质数，合数，约数，倍数
- ⑤ 同余方程
- ⑥ 欧拉函数
 - 积性函数，线性筛
 - 欧拉定理/费马小定理

求 $a^b \bmod p$ 的值, $a, b, p \leq 10^9$ 。
——我会! 不就是快速幂吗?

求 $a^b \bmod p$ 的值, $a, b, p \leq 10^9$ 。

——我会! 不就是快速幂吗?

求 $a^b \bmod p$ 的值, $a, b \leq 10^{10000000}, p \leq 10^9$ 。

——????????

若 p 为质数，则对任意 $1 \leq a \leq p-1$ ，均有 $a^{p-1} \bmod p = 1$ 。

若 p 为质数，则对任意 $1 \leq a \leq p-1$ ，均有 $a^{p-1} \bmod p = 1$ 。

推论：

对任意 $1 \leq a \leq p-1, b \geq 0$ ，均有 $a^b \bmod p = a^{b \bmod (p-1)} \bmod p$ 。

对任意 $1 \leq a \leq m-1, (a, m) = 1$, 均有 $a^{\varphi(m)} \bmod m = 1$ 。

对任意 $1 \leq a \leq m-1, (a, m) = 1$, 均有 $a^{\varphi(m)} \bmod m = 1$ 。

推论:

对任意 $1 \leq a \leq m-1, (a, m) = 1, b \geq 0$, 均有
 $a^b \bmod m = a^{b \bmod \varphi(m)} \bmod m$ 。

事实上费马小定理可以看做欧拉定理的特殊情况。

对任意 $1 \leq a \leq m-1, (a, m) > 1, b \geq \varphi(m)$, 均有

$$a^b \bmod m = a^{b \bmod \varphi(m) + \varphi(m)} \bmod m。$$

设 $1 \leq a < m$, 若存在 $1 \leq b < m$ 满足 $ab \bmod m = 1$, 则称 b 是 a 在模 m 意义下的乘法逆元, 简称逆元, 记作 a^{-1} 。

设 $1 \leq a < m$ ，若存在 $1 \leq b < m$ 满足 $ab \bmod m = 1$ ，则称 b 是 a 在模 m 意义下的乘法逆元，简称逆元，记作 a^{-1} 。

若 $(a, m) = d > 1$ ，则 $d | (ab \bmod m)$ ，因此 a 存在逆元的充要条件是 $(a, m) = 1$ 。

设 $1 \leq a < m$ ，若存在 $1 \leq b < m$ 满足 $ab \bmod m = 1$ ，则称 b 是 a 在模 m 意义下的乘法逆元，简称逆元，记作 a^{-1} 。

若 $(a, m) = d > 1$ ，则 $d | (ab \bmod m)$ ，因此 a 存在逆元的充要条件是 $(a, m) = 1$ 。

根据欧拉定理， $a^{-1} = a^{\varphi(m)-1} \bmod m$ 。

设 $1 \leq a < m$ ，若存在 $1 \leq b < m$ 满足 $ab \bmod m = 1$ ，则称 b 是 a 在模 m 意义下的乘法逆元，简称逆元，记作 a^{-1} 。

若 $(a, m) = d > 1$ ，则 $d | (ab \bmod m)$ ，因此 a 存在逆元的充要条件是 $(a, m) = 1$ 。

根据欧拉定理， $a^{-1} = a^{\varphi(m)-1} \bmod m$ 。

如果 p 为质数，则 $a^{-1} = a^{p-2} \bmod p$ 。

设 $1 \leq a < m$ ，若存在 $1 \leq b < m$ 满足 $ab \bmod m = 1$ ，则称 b 是 a 在模 m 意义下的乘法逆元，简称逆元，记作 a^{-1} 。

若 $(a, m) = d > 1$ ，则 $d | (ab \bmod m)$ ，因此 a 存在逆元的充要条件是 $(a, m) = 1$ 。

根据欧拉定理， $a^{-1} = a^{\varphi(m)-1} \bmod m$ 。

如果 p 为质数，则 $a^{-1} = a^{p-2} \bmod p$ 。

因为这种做法要求出欧拉函数，当 m 较大（超过 10^9 ）时复杂度过高。所以当 m 不是质数时，我们一般用下一节讲的扩展欧几里得算法（exgcd）去求逆元。

当我们需要的逆元都较小而又很多时，可以用线性逆元一次性求出 $1 \sim n$ 的所有逆元。

设 $p = k \cdot i + r, 1 < i < p, r < i$ ，则 $k = \lfloor \frac{p}{i} \rfloor$

$$k \cdot i + r = 0 \pmod{p}$$

两边同时乘 $i^{-1}r^{-1}$ ，则

$$k \cdot r^{-1} + i^{-1} = 0 \pmod{p}$$

故 $i^{-1} = -k \cdot r^{-1} = -\lfloor \frac{p}{i} \rfloor r^{-1}$

因为 r^{-1} 已经求出，所以可以递推求出 `inv`。

```
inv[1]=1;
```

```
for(int i=2;i<=n;++i) inv[i]=inv[mod%i]*(mod-mod/i)%mod;
```

Thanks for listening!

Any Questions?

关于下午考试：祝大家好运。