# 数学专题

neuacm-2024Training

**Pujx**

东北大学 计算机科学与工程学院

2024 年 8 月 5 日


# 整除

- 如果存在某个整数 $k$, 使得 $a = k \cdot d$, 则称 $d \mid a$ ($d$ 整除 $a$).

# 整除

- 如果存在某个整数 $k$, 使得 $a = k \cdot d$, 则称 $d \mid a$ ($d$ 整除 $a$).
- $\mid$ 是整除符号.

# 整除

- 如果存在某个整数 $k$, 使得 $a = k \cdot d$, 则称 $d \mid a$ ($d$ 整除 $a$).
- $\mid$ 是整除符号.
- $d$ 是 $a$ 的约数, $a$ 是 $d$ 的倍数.

# 整除

- 如果存在某个整数 $k$, 使得 $a = k \cdot d$, 则称 $d \mid a$ ($d$ 整除 $a$).
- $\mid$ 是整除符号.
- $d$ 是 $a$ 的约数, $a$ 是 $d$ 的倍数.

## 整除的性质

# 整除

- 如果存在某个整数 $k$, 使得 $a = k \cdot d$, 则称 $d \mid a$ ($d$ 整除 $a$).
- $\mid$ 是整除符号.
- $d$ 是 $a$ 的约数, $a$ 是 $d$ 的倍数.

## 整除的性质

- $a \mid b \Leftrightarrow -a \mid b, a \mid -b, -a \mid -b, |a| \mid |b|$.

# 整除

- 如果存在某个整数 $k$, 使得 $a = k \cdot d$, 则称 $d \mid a$ ($d$ 整除 $a$).
- $\mid$ 是整除符号.
- $d$ 是 $a$ 的约数, $a$ 是 $d$ 的倍数.

## 整除的性质

- $a \mid b \Leftrightarrow -a \mid b, a \mid -b, -a \mid -b, |a| \mid |b|$.
- $a \mid b, b \mid c \Rightarrow a \mid c$.

# 整除

- 如果存在某个整数 $k$, 使得 $a = k \cdot d$, 则称 $d \mid a$ ($d$ 整除 $a$).
- $\mid$ 是整除符号.
- $d$ 是 $a$ 的约数, $a$ 是 $d$ 的倍数.

## 整除的性质

- $a \mid b \Leftrightarrow -a \mid b, a \mid -b, -a \mid -b, |a| \mid |b|$.
- $a \mid b, b \mid c \Rightarrow a \mid c$.
- $a \mid b, a \mid c \Leftrightarrow \forall x, y \in \mathbb{Z}, a \mid (bx + cy)$.

# 整除

- 如果存在某个整数 $k$, 使得 $a = k \cdot d$, 则称 $d \mid a$ ($d$ 整除 $a$).
- $\mid$ 是整除符号.
- $d$ 是 $a$ 的约数, $a$ 是 $d$ 的倍数.

## 整除的性质

- $a \mid b \Leftrightarrow -a \mid b, a \mid -b, -a \mid -b, |a| \mid |b|$.
- $a \mid b, b \mid c \Rightarrow a \mid c$.
- $a \mid b, a \mid c \Leftrightarrow \forall x, y \in \mathbb{Z}, a \mid (bx + cy)$.
- $m \neq 0, a \mid b \Leftrightarrow ma \mid mb$.

# 整除

- 如果存在某个整数 $k$, 使得 $a = k \cdot d$, 则称 $d \mid a$ ($d$ 整除 $a$).
- $\mid$ 是整除符号.
- $d$ 是 $a$ 的约数, $a$ 是 $d$ 的倍数.

## 整除的性质

- $a \mid b \Leftrightarrow -a \mid b, a \mid -b, -a \mid -b, |a| \mid |b|$.
- $a \mid b, b \mid c \Rightarrow a \mid c$.
- $a \mid b, a \mid c \Leftrightarrow \forall x, y \in \mathbb{Z}, a \mid (bx + cy)$.
- $m \neq 0, a \mid b \Leftrightarrow ma \mid mb$.
- $a \mid b, b \mid a \Leftrightarrow b = \pm a$.

# 质数 (素数)

- 大于 1 的自然数中, 除了 1 和本身以外, 不再有其他因数的数.

# 质数 (素数)

- 大于 1 的自然数中, 除了 1 和本身以外, 不再有其他因数的数.
- 合数: 大于 1 的自然数中, 除了能被 1 和本身整除外，还能被其他数整除的数.

# 质数 (素数)

- 大于 1 的自然数中, 除了 1 和本身以外, 不再有其他因数的数.
- 合数: 大于 1 的自然数中, 除了能被 1 和本身整除外，还能被其他数整除的数.
- 1 既不是质数, 也不是合数.

# 唯一分解定理 (算数基本定理)

## 唯一分解定理

对于 $\forall a \in \mathbb{Z}, a > 1$, 能够唯一地写成 $a = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, 其中 $p_i$ 是质数, $\alpha_i > 0$, $i = 1, \cdots, k$, 且 $p_i < p_j \, (i < j)$.

# 唯一分解定理 (算数基本定理)

## 唯一分解定理

对于 $\forall a \in \mathbb{Z}, a > 1$, 能够唯一地写成 $a = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, 其中 $p_i$ 是质数, $\alpha_i > 0$, $i = 1, \cdots, k$, 且 $p_i < p_j\,(i < j)$.

- 若 $a$ 为合数, 则 $\exists p \mid a, p \leqslant \sqrt{a}$.

# 唯一分解定理 (算数基本定理)

## 唯一分解定理

对于 $\forall a \in \mathbb{Z}, a > 1$, 能够唯一地写成 $a = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, 其中 $p_i$ 是质数, $\alpha_i > 0$, $i = 1, \cdots, k$, 且 $p_i < p_j \, (i < j)$.

- 若 $a$ 为合数, 则 $\exists p \mid a, p \leqslant \sqrt{a}$.
- 判断 $n$ 是否为质数: 枚举 $1 \sim \sqrt{n}$ 判断是否为因数, 复杂度 $\mathcal{O}(\sqrt{n})$.

# 唯一分解定理 (算数基本定理)

## 唯一分解定理

对于 $\forall a \in \mathbb{Z}, a > 1$, 能够唯一地写成 $a = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, 其中 $p_i$ 是质数, $\alpha_i > 0$, $i = 1, \cdots, k$, 且 $p_i < p_j\,(i < j)$.

- 若 $a$ 为合数, 则 $\exists p \mid a, p \leqslant \sqrt{a}$.
- 判断 $n$ 是否为质数: 枚举 $1 \sim \sqrt{n}$ 判断是否为因数, 复杂度 $\mathcal{O}(\sqrt{n})$.
- 因数个数公式: $d(n) = \prod_{i=1}^{k} (\alpha_i + 1)$.

# 唯一分解定理 (算数基本定理)

## 唯一分解定理

对于 $\forall a \in \mathbb{Z}, a > 1$, 能够唯一地写成 $a = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, 其中 $p_i$ 是质数, $\alpha_i > 0$, $i = 1, \cdots, k$, 且 $p_i < p_j \, (i < j)$.

- 若 $a$ 为合数, 则 $\exists p \mid a, p \leqslant \sqrt{a}$.
- 判断 $n$ 是否为质数: 枚举 $1 \sim \sqrt{n}$ 判断是否为因数, 复杂度 $\mathcal{O}(\sqrt{n})$.
- 因数个数公式: $d(n) = \prod_{i=1}^{k} (\alpha_i + 1)$.
- 因数和公式: $\sigma(n) = \prod_{i=1}^{k} \sum_{j=0}^{\alpha_i} p_i^j$.

# 唯一分解定理 (算数基本定理)

## 唯一分解定理

对于 $\forall a \in \mathbb{Z}, a > 1$, 能够唯一地写成 $a = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, 其中 $p_i$ 是质数, $\alpha_i > 0$, $i = 1, \cdots, k$, 且 $p_i < p_j\,(i < j)$.

## Question.1

如何证明质数的个数是无穷的?

# 唯一分解定理 (算数基本定理)

## 唯一分解定理

对于 $\forall a \in \mathbb{Z}, a > 1$, 能够唯一地写成 $a = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, 其中 $p_i$ 是质数, $\alpha_i > 0$, $i = 1, \cdots, k$, 且 $p_i < p_j\,(i < j)$.

## Question.1

如何证明质数的个数是无穷的?

## Question.2

如何估算小于等于 $n$ 的质数个数 $\pi(n)$? (自行上网查阅)

# 唯一分解定理 (算数基本定理)

## 唯一分解定理

对于 $\forall a \in \mathbb{Z}, a > 1$, 能够唯一地写成 $a = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, 其中 $p_i$ 是质数, $\alpha_i > 0$, $i = 1, \cdots, k$, 且 $p_i < p_j\,(i < j)$.

## Question.1

如何证明质数的个数是无穷的?

## Question.2

如何估算小于等于 $n$ 的质数个数 $\pi(n)$? (自行上网查阅)

$\pi(n)$ 的下界为 $\log_2(\log_2 n)$, $\pi(n)$ 的上界为 $\dfrac{n}{\ln n}$.

# 质数筛法

**问题**

求 1 到 $n$ 内的所有质数. $1 \leqslant n \leqslant 10^6$.

# 质数筛法

**问题**

求 1 到 $n$ 内的所有质数. $1 \leqslant n \leqslant 10^6$.

- 考虑朴素算法, 即逐个判断每个数是否为质数.

# 质数筛法

**问题**

求 1 到 $n$ 内的所有质数. $1 \leqslant n \leqslant 10^6$.

- 考虑朴素算法, 即逐个判断每个数是否为质数.
- 时间复杂度 $\sum_{i=1}^{n} \sqrt{i} = \mathcal{O}(n\sqrt{n})$.

# 质数筛法

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 |
| 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 |
| 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 |
| 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 |
| 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 |
| 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 |

# 质数筛法

## 埃氏筛 (Sieve of Eratosthenes)

用质数把质数的倍数筛掉.

# 质数筛法

## 埃氏筛 (Sieve of Eratosthenes)

用质数把质数的倍数筛掉.

```
void sieve(int n) {
    for (int i = 2; i <= n; i++)
        if (!not_prime[i]) {
            prime[++tot] = i;
            for (int j = 2 * i; j <= n; j += i)
                not_prime[j] = 1;
        }
}
```

# 质数筛法

## 埃氏筛 (Sieve of Eratosthenes)

用质数把质数的倍数筛掉.

```
void sieve(int n) {
    for (int i = 2; i <= n; i++)
        if (!not_prime[i]) {
            prime[++tot] = i;
            for (int j = 2 * i; j <= n; j += i)
                not_prime[j] = 1;
        }
}
```

- 时间复杂度: $\mathcal{O}(n \log(\log n))$ (Mertens' 2nd theorem).

# 质数筛法

## 埃氏筛 (Sieve of Eratosthenes)

用质数把质数的倍数筛掉.

```
void sieve(int n) {
    for (int i = 2; i <= n; i++)
        if (!not_prime[i]) {
            prime[++tot] = i;
            for (int j = 2 * i; j <= n; j += i)
                not_prime[j] = 1;
        }
}
```

- 时间复杂度: $\mathcal{O}(n \log(\log n))$ (Mertens' 2nd theorem).
- 改进思路: 有些数会被多个因子筛除, 例如 $6$, 会被 $2, 3$ 各筛一次.

# 质数筛法

## 欧拉筛 (Euler Sieve)

每个合数只需要被其最小的质因子筛掉.

# 质数筛法

## 欧拉筛 (Euler Sieve)

每个合数只需要被其最小的质因子筛掉.

```
void sieve(int n) {
    for (int i = 2; i <= n; i++) {
        if (!not_prime[i]) prime[++tot] = i;
        for (int j = 1; j <= tot && i * prime[j] <= n; j++) {
            not_prime[i * prime[j]] = 1;
            if (i % prime[j] == 0) break;
        }
    }
}
```

# 质数筛法

## 欧拉筛 (Euler Sieve)

每个合数只需要被其最小的质因子筛掉.

```
void sieve(int n) {
    for (int i = 2; i <= n; i++) {
        if (!not_prime[i]) prime[++tot] = i;
        for (int j = 1; j <= tot && i * prime[j] <= n; j++) {
            not_prime[i * prime[j]] = 1;
            if (i % prime[j] == 0) break;
        }
    }
}
```

- 时间复杂度: $\mathcal{O}(n)$.

# 质数筛法

## Prime Distance (POJ 2689)

求 $[L, R]$ 区间中, 距离最近的一对和最远的一对质数. 多组询问.
$1 \leqslant L < R \leqslant 2147483647, \sum(R - L + 1) \leqslant 10^6$.

# GCD 和 LCM

- $d \in \mathbb{Z}, a_1, a_2 \in \mathbb{Z}$.

# GCD 和 LCM

- $d \in \mathbb{Z}, a_1, a_2 \in \mathbb{Z}$.
- 如果 $d \mid a_1, d \mid a_2$, 则称 $d$ 为 $a_1, a_2$ 的公因数.

# GCD 和 LCM

- $d \in \mathbb{Z}, a_1, a_2 \in \mathbb{Z}$.
- 如果 $d \mid a_1, d \mid a_2$, 则称 $d$ 为 $a_1, a_2$ 的公因数.
- 如果 $a_1 \mid d, a_2 \mid d$, 则称 $d$ 为 $a_1, a_2$ 的公倍数.

# GCD 和 LCM

- $d \in \mathbb{Z}, a_1, a_2 \in \mathbb{Z}$.
- 如果 $d \mid a_1, d \mid a_2$, 则称 $d$ 为 $a_1, a_2$ 的公因数.
- 如果 $a_1 \mid d, a_2 \mid d$, 则称 $d$ 为 $a_1, a_2$ 的公倍数.
- $a_1, a_2$ 的所有公因数中最大的称为最大公因数 (Greatest Common Divisor), 记为 $(a_1, a_2)$.

# GCD 和 LCM

- $d \in \mathbb{Z}, a_1, a_2 \in \mathbb{Z}$.
- 如果 $d \mid a_1, d \mid a_2$, 则称 $d$ 为 $a_1, a_2$ 的公因数.
- 如果 $a_1 \mid d, a_2 \mid d$, 则称 $d$ 为 $a_1, a_2$ 的公倍数.
- $a_1, a_2$ 的所有公因数中最大的称为最大公因数 (Greatest Common Divisor), 记为 $(a_1, a_2)$.
- $a_1, a_2$ 的所有公倍数中最小的称为最小公倍数 (Leatest Common Multiple), 记为 $[a_1, a_2]$.

# GCD 和 LCM

- $d \in \mathbb{Z}, a_1, a_2 \in \mathbb{Z}$.
- 如果 $d \mid a_1, d \mid a_2$, 则称 $d$ 为 $a_1, a_2$ 的公因数.
- 如果 $a_1 \mid d, a_2 \mid d$, 则称 $d$ 为 $a_1, a_2$ 的公倍数.
- $a_1, a_2$ 的所有公因数中最大的称为最大公因数 (Greatest Common Divisor), 记为 $(a_1, a_2)$.
- $a_1, a_2$ 的所有公倍数中最小的称为最小公倍数 (Leatest Common Multiple), 记为 $[a_1, a_2]$.
- $[a_1, a_2] = \dfrac{a_1 a_2}{(a_1, a_2)}$.

# GCD 和 LCM

- $d \in \mathbb{Z}, a_1, a_2 \in \mathbb{Z}$.
- 如果 $d \mid a_1, d \mid a_2$, 则称 $d$ 为 $a_1, a_2$ 的公因数.
- 如果 $a_1 \mid d, a_2 \mid d$, 则称 $d$ 为 $a_1, a_2$ 的公倍数.
- $a_1, a_2$ 的所有公因数中最大的称为最大公因数 (Greatest Common Divisor), 记为 $(a_1, a_2)$.
- $a_1, a_2$ 的所有公倍数中最小的称为最小公倍数 (Leatest Common Multiple), 记为 $[a_1, a_2]$.
- $[a_1, a_2] = \dfrac{a_1 a_2}{(a_1, a_2)}$.
- $(a_1, a_2) = 1 \Leftrightarrow a_1, a_2$ 互质.

# 欧几里得算法

## 问题

给定整数 $a, b$, 计算 $(a, b)$. $1 \leqslant a, b \leqslant 10^{18}$.

# 欧几里得算法

## 问题

给定整数 $a, b$, 计算 $(a, b)$. $1 \leqslant a, b \leqslant 10^{18}$.

- 朴素算法: 从大到小枚举每个数, 判断是否为 GCD. $\mathcal{O}(\min\{a, b\})$.

# 欧几里得算法

## 问题

给定整数 $a, b$, 计算 $(a, b)$. $1 \leqslant a, b \leqslant 10^{18}$.

- 朴素算法: 从大到小枚举每个数, 判断是否为 GCD. $\mathcal{O}(\min\{a, b\})$.

## 辗转相减法

$(a, b) = (b, a - b)$.

# 欧几里得算法

## 问题

给定整数 $a, b$, 计算 $(a, b)$. $1 \leqslant a, b \leqslant 10^{18}$.

## 欧几里得算法 (辗转相除法)

$(a, b) = (b, a \% b)$.

# 欧几里得算法

## 问题

给定整数 $a, b$, 计算 $(a, b)$. $1 \leqslant a, b \leqslant 10^{18}$.

## 欧几里得算法 (辗转相除法)

$(a, b) = (b, a \% b)$.

```
int gcd(int a, int b) {
    return !b ? a : gcd(b, a % b);
}
```

# 欧几里得算法

## 问题

给定整数 $a, b$, 计算 $(a, b)$. $1 \leqslant a, b \leqslant 10^{18}$.

## 欧几里得算法 (辗转相除法)

$(a, b) = (b, a \% b)$.

```
int gcd(int a, int b) {
    return !b ? a : gcd(b, a % b);
}
```

- 时间复杂度: $\mathcal{O}(\log(\min\{a, b\}))$.

# 扩展欧几里得算法 (exgcd)

## 裴蜀定理

对任意整数 $a, b$ ($a, b$ 不全为 0), 一定存在整数 $x, y$, 使不定方程 $ax + by = (a, b)$ 成立. (证明略)

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

- $ax + by = (a, b)$,

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

- $ax + by = (a, b)$,
- $bx' + (a \% b)y' = (b, a \% b)$,

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

- $ax + by = (a, b)$,
- $bx' + (a \% b)y' = (b, a \% b)$,
- 由 $(a, b) = (b, a \% b)$, $a \% b = a - \left\lfloor \frac{a}{b} \right\rfloor b$,

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

- $ax + by = (a, b)$,
- $bx' + (a \% b)y' = (b, a \% b)$,
- 由 $(a, b) = (b, a \% b)$, $a \% b = a - \left\lfloor \frac{a}{b} \right\rfloor b$,
- 可得 $bx' + \left(a - \left\lfloor \frac{a}{b} \right\rfloor b\right) y' = (a, b)$,

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

- $ax + by = (a, b)$,
- $bx' + (a \% b)y' = (b, a \% b)$,
- 由 $(a, b) = (b, a \% b)$, $a \% b = a - \left\lfloor \frac{a}{b} \right\rfloor b$,
- 可得 $bx' + \left(a - \left\lfloor \frac{a}{b} \right\rfloor b\right) y' = (a, b)$,
- 整理得 $ay' + b\left(x' - \left\lfloor \frac{a}{b} \right\rfloor y'\right) = (a, b)$,

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

- $ax + by = (a, b)$,
- $bx' + (a \% b)y' = (b, a \% b)$,
- 由 $(a, b) = (b, a \% b)$, $a \% b = a - \left\lfloor \frac{a}{b} \right\rfloor b$,
- 可得 $bx' + \left(a - \left\lfloor \frac{a}{b} \right\rfloor b\right) y' = (a, b)$,
- 整理得 $ay' + b\left(x' - \left\lfloor \frac{a}{b} \right\rfloor y'\right) = (a, b)$,
- 带回原方程可得 $x = y', y = x' - \left\lfloor \frac{a}{b} \right\rfloor y'$, 递归求解即可.

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

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

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

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

- 时间复杂度: $\mathcal{O}(\log(\min\{a, b\}))$.

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 0), 求不定方程 $ax + by = (a, b)$ 的整数解.

- 求出的为方程的特解 $(x_0, y_0)$, 通解为

$$\left(x_0 + k\frac{b}{(a,b)}, y_0 - k\frac{a}{(a,b)}\right).$$

# 扩展欧几里得算法 (exgcd)

## 问题

给定整数 $a, b$ ($a, b$ 不全为 $0$), 求不定方程 $ax + by = (a, b)$ 的整数解.

- 求出的为方程的特解 $(x_0, y_0)$, 通解为

$$\left(x_0 + k\frac{b}{(a,b)}, y_0 - k\frac{a}{(a,b)}\right).$$

- $ax + by = d$ 当且仅当 $(a, b) \mid d$. 可转换为方程 $ax' + by' = (a, b), k = \frac{d}{(a,b)}, x' = \frac{x}{k}, y' = \frac{y}{k}$ 计算.

# 扩展欧几里得算法 (exgcd)

## 青蛙的约会 (POJ 1061)

周长为 $L$ 的圆, 坐标为 $[0, L-1]$.
两只青蛙在圆上, 初始坐标为 $x, y$, 都向顺时针方向跳, 每次跳动的距离为 $m, n$, 求他们第一次相遇时跳了多少次, 或者不能相遇输出 `Impossible`.
$x \neq y < 2 \times 10^9, 0 < m, n < 2 \times 10^9, 0 < L < 2.1 \times 10^9$.

# 同余

- $a, b$ 两个整数, 对于一个正模数 $m$, 如果满足 $a \% m = b \% m$, 则称 $a$ 与 $b$ 对模 $m$ 同余, 记为 $a \equiv b \pmod{m}$.

# 同余

- $a, b$ 两个整数, 对于一个正模数 $m$, 如果满足 $a \% m = b \% m$, 则称 $a$ 与 $b$ 对模 $m$ 同余, 记为 $a \equiv b \pmod{m}$.

## 同余的性质

# 同余

- $a, b$ 两个整数, 对于一个正模数 $m$, 如果满足 $a \% m = b \% m$, 则称 $a$ 与 $b$ 对模 $m$ 同余, 记为 $a \equiv b \pmod{m}$.

## 同余的性质

- $a \equiv a \pm m \pmod{m}$.

# 同余

- $a, b$ 两个整数, 对于一个正模数 $m$, 如果满足 $a \% m = b \% m$, 则称 $a$ 与 $b$ 对模 $m$ 同余, 记为 $a \equiv b \pmod{m}$.

## 同余的性质

- $a \equiv a \pm m \pmod{m}$.
- $a \pm b \equiv (a \mod m) \pm (b \mod m) \pmod{m}$.

# 同余

- $a, b$ 两个整数, 对于一个正模数 $m$, 如果满足 $a \% m = b \% m$, 则称 $a$ 与 $b$ 对模 $m$ 同余, 记为 $a \equiv b \pmod{m}$.

## 同余的性质

- $a \equiv a \pm m \pmod{m}$.
- $a \pm b \equiv (a \mod m) \pm (b \mod m) \pmod{m}$.
- $a \times b \equiv (a \mod m) \times (b \mod m) \pmod{m}$.

# 同余

- $a, b$ 两个整数, 对于一个正模数 $m$, 如果满足 $a \% m = b \% m$, 则称 $a$ 与 $b$ 对模 $m$ 同余, 记为 $a \equiv b \pmod{m}$.

## 同余的性质

- $a \equiv a \pm m \pmod{m}$.
- $a \pm b \equiv (a \mod m) \pm (b \mod m) \pmod{m}$.
- $a \times b \equiv (a \mod m) \times (b \mod m) \pmod{m}$.
- $a$ 是 $b$ 的倍数 $\Leftrightarrow a \equiv 0 \pmod{b}$.

# 同余

- $a, b$ 两个整数, 对于一个正模数 $m$, 如果满足 $a \% m = b \% m$, 则称 $a$ 与 $b$ 对模 $m$ 同余, 记为 $a \equiv b \pmod{m}$.

## 同余的性质

- $a \equiv a \pm m \pmod{m}$.
- $a \pm b \equiv (a \mod m) \pm (b \mod m) \pmod{m}$.
- $a \times b \equiv (a \mod m) \times (b \mod m) \pmod{m}$.
- $a$ 是 $b$ 的倍数 $\Leftrightarrow a \equiv 0 \pmod{b}$.

- 对于只含有 $+, -, \times$ 的运算, 可以在任何时刻对其中的数取模, 其结果和原来的式子都是同余的.

# 同余

- $a, b$ 两个整数, 对于一个正模数 $m$, 如果满足 $a \% m = b \% m$, 则称 $a$ 与 $b$ 对模 $m$ 同余, 记为 $a \equiv b \pmod{m}$.

## 同余的性质

- $a \equiv a \pm m \pmod{m}$.
- $a \pm b \equiv (a \mod m) \pm (b \mod m) \pmod{m}$.
- $a \times b \equiv (a \mod m) \times (b \mod m) \pmod{m}$.
- $a$ 是 $b$ 的倍数 $\Leftrightarrow a \equiv 0 \pmod{b}$.

- 对于只含有 $+, -, \times$ 的运算, 可以在任何时刻对其中的数取模, 其结果和原来的式子都是同余的.
- $a \div b \equiv (a \mod m) \div (b \mod m) \pmod{m}$ ?

# 快速幂

## 问题

计算 $a^b \mod m$, $0 \leqslant b \leqslant 10^{18}$.

# 快速幂

## 问题

计算 $a^b \mod m$, $0 \leqslant b \leqslant 10^{18}$.

- 朴素算法: $\mathcal{O}(b)$, 代价太高.

# 快速幂

## 问题

计算 $a^b \mod m$, $0 \leqslant b \leqslant 10^{18}$.

- 朴素算法: $\mathcal{O}(b)$, 代价太高.
- 快速幂: 二进制取幂.

# 快速幂

## 问题

计算 $a^b \mod m$, $0 \leqslant b \leqslant 10^{18}$.

- 朴素算法: $\mathcal{O}(b)$, 代价太高.
- 快速幂: 二进制取幂.
- 以 $5^{27}$ 为例, $27 = (11011)_2$, $5^{27} = 5^1 \times 5^2 \times 5^8 \times 5^{16}$.

# 快速幂

## 问题

计算 $a^b \mod m$, $0 \leqslant b \leqslant 10^{18}$.

- 朴素算法: $\mathcal{O}(b)$, 代价太高.
- 快速幂: 二进制取幂.
- 以 $5^{27}$ 为例, $27 = (11011)_2$, $5^{27} = 5^1 \times 5^2 \times 5^8 \times 5^{16}$.

```
int ksm(int a, int b, int m) {
    int ans = 1;
    for (; b; a = 1ll * a * a % m, b >>= 1)
        if (b & 1) ans = 1ll * ans * a % m;
    return ans;
}
```

# 快速幂

## 问题

计算 $a^b \mod m$, $0 \leqslant b \leqslant 10^{18}$.

- 朴素算法: $\mathcal{O}(b)$, 代价太高.
- 快速幂: 二进制取幂.
- 以 $5^{27}$ 为例, $27 = (11011)_2$, $5^{27} = 5^1 \times 5^2 \times 5^8 \times 5^{16}$.

```
int ksm(int a, int b, int m) {
    int ans = 1;
    for (; b; a = 1ll * a * a % m, b >>= 1)
        if (b & 1) ans = 1ll * ans * a % m;
    return ans;
}
```

- 时间复杂度: $\mathcal{O}(\log n)$.

# 逆元

- $a \div b$ 在同余式中应当如何计算?

# 逆元

- $a \div b$ 在同余式中应当如何计算?

**逆元**

如果 $a, b \in \mathbb{Z}_m$, 满足 $ab \equiv 1 \pmod{m}$, 则称 $b$ 是 $a$ 的逆元, 记作 $a^{-1}$.

# 逆元

- $a \div b$ 在同余式中应当如何计算?

**逆元**

如果 $a, b \in \mathbb{Z}_m$, 满足 $ab \equiv 1 \pmod{m}$, 则称 $b$ 是 $a$ 的逆元, 记作 $a^{-1}$.

- 如何判定在 $\mathbb{Z}_m$ 中, $a$ 是否有逆元? 如果有, 如何计算 $a^{-1}$?

# 逆元计算方法 1

- $ab = km + 1$, 扩展欧几里得算法 (exgcd).

# 逆元计算方法 1

- $ab = km + 1$, 扩展欧几里得算法 (exgcd).
- 有逆元 (不定方程有解) 当且仅当 $(a, m) = 1$.

# 逆元计算方法 1

- $ab = km + 1$, 扩展欧几里得算法 (exgcd).
- 有逆元 (不定方程有解) 当且仅当 $(a, m) = 1$.
- 如果 $m$ 为质数, 则任意 $0 < a < m$, $a$ 均有逆元.

# 逆元计算方法 2

**费马小定理**

设 $p$ 为质数, 如果 $p \nmid a$, 那么 $a^{p-1} \equiv 1 \pmod{p}$.

# 逆元计算方法 2

**费马小定理**

设 $p$ 为质数, 如果 $p \nmid a$, 那么 $a^{p-1} \equiv 1 \pmod{p}$.

- 设 $p$ 为质数, $a \in \mathbb{Z}_p$, $0 < a < p$, 则有 $a^{-1} \equiv a^{p-2} \pmod{p}$.

# 逆元计算方法 2

**费马小定理**

设 $p$ 为质数, 如果 $p \nmid a$, 那么 $a^{p-1} \equiv 1 \pmod{p}$.

- 设 $p$ 为质数, $a \in \mathbb{Z}_p$, $0 < a < p$, 则有 $a^{-1} \equiv a^{p-2} \pmod{p}$.
- 快速幂实现, 时间复杂度 $\mathcal{O}(\log n)$.

# 逆元计算方法 2

**费马小定理**

设 $p$ 为质数, 如果 $p \nmid a$, 那么 $a^{p-1} \equiv 1 \pmod{p}$.

- 设 $p$ 为质数, $a \in \mathbb{Z}_p$, $0 < a < p$, 则有 $a^{-1} \equiv a^{p-2} \pmod{p}$.
- 快速幂实现, 时间复杂度 $\mathcal{O}(\log n)$.
- 缺点: 模数限制较大, 只能为质数.

# 欧拉函数

## 欧拉函数

欧拉函数 $\varphi(n)$ 为正整数 $n$ 与序列 $1, 2, \cdots, n-1, n$ 中互质的数的个数. 即 $\varphi(n) = \sum_{i=1}^{n} [(i, n) = 1]$.

# 欧拉函数

## 欧拉函数

欧拉函数 $\varphi(n)$ 为正整数 $n$ 与序列 $1, 2, \cdots, n-1, n$ 中互质的数的个数. 即 $\varphi(n) = \sum_{i=1}^{n} [(i, n) = 1]$.

- 设 $n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, 则

$$\varphi(n) = \prod_{i=1}^{k} (p_i - 1) p_i^{\alpha_i - 1}.$$

# 欧拉函数

- 求单点的欧拉函数值 $\varphi(n)$. 在求质因子的基础上略加修改.

# 欧拉函数

- 求单点的欧拉函数值 $\varphi(n)$. 在求质因子的基础上略加修改.

```
int get_phi(int n) {
    int phi = 1;
    for (int i = 2; i * i <= n; i++) {
        if (n % i == 0) {
            phi *= (i - 1), n /= i;
            while (n % i == 0) phi *= i, n /= i;
        }
    }
    if (n > 1) phi *= (n - 1);
    return phi;
}
```

# 欧拉函数

- 求 $\varphi(1), \varphi(2), \cdots, \varphi(n)$ 的值. 在欧拉筛的基础上略加修改.

# 欧拉函数

- 求 $\varphi(1), \varphi(2), \cdots, \varphi(n)$ 的值. 在欧拉筛的基础上略加修改.

```
void init_phi(int n) {
    phi[1] = 1;
    for (int i = 2; i <= n; i++) {
        if (!not_prime[i]) prime[++tot] = i, phi[i] = i - 1;
        for (int j = 1; j <= tot && i * prime[j] <= n; j++) {
            not_prime[i * prime[j]] = 1;
            if (i % prime[j] == 0) {
                phi[i * prime[j]] = phi[i] * prime[j];
                break;
            }
            phi[i * prime[j]] = phi[i] * (prime[j] - 1);
        }
    }
}
```

# 逆元计算方法 3

**欧拉定理**

如果 $(a, m) = 1$, 那么 $a^{\varphi(m)} \equiv 1 \pmod{m}$.

# 逆元计算方法 3

**欧拉定理**

如果 $(a, m) = 1$, 那么 $a^{\varphi(m)} \equiv 1 \pmod{m}$.

- 设 $m$ 为正整数, $a \in \mathbb{Z}_m$, $(a, m) = 1$, 则有 $a^{-1} \equiv a^{\varphi(m)-1} \pmod{m}$.

# 逆元计算方法 4

## 问题

计算 1 到 $n$ 在模数为质数 $p$ 意义下的逆元. $1 \leqslant n \leqslant 10^7$.

# 逆元计算方法 4

**问题**

计算 1 到 $n$ 在模数为质数 $p$ 意义下的逆元. $1 \leqslant n \leqslant 10^7$.

- $\left\lfloor \frac{p}{i} \right\rfloor i + p \% i = p$

# 逆元计算方法 4

**问题**

计算 1 到 $n$ 在模数为质数 $p$ 意义下的逆元. $1 \leqslant n \leqslant 10^7$.

- $\left\lfloor \frac{p}{i} \right\rfloor i + p \% i = p$
- $\Rightarrow \left\lfloor \frac{p}{i} \right\rfloor i + p \% i \equiv 0 \pmod{p}$

# 逆元计算方法 4

**问题**

计算 1 到 $n$ 在模数为质数 $p$ 意义下的逆元. $1 \leqslant n \leqslant 10^7$.

- $\left\lfloor \frac{p}{i} \right\rfloor i + p \% i = p$
- $\Rightarrow \left\lfloor \frac{p}{i} \right\rfloor i + p \% i \equiv 0 \pmod{p}$
- $\Rightarrow i^{-1} \equiv -\left\lfloor \frac{p}{i} \right\rfloor (p \% i)^{-1} \pmod{p}$.

# 中国剩余定理 (CRT)

**《孙子算经》**

今有物不知其数，三三数之剩二，五五数之剩三，七七数之剩二，问物几何？

# 中国剩余定理 (CRT)

**《孙子算经》**

今有物不知其数，三三数之剩二，五五数之剩三，七七数之剩二，问物几何？

- 答案：23.

# 中国剩余定理 (CRT)

**《孙子算经》**

今有物不知其数, 三三数之剩二, 五五数之剩三, 七七数之剩二, 问物几何?

- 答案: 23.
- $x \equiv 23 \pmod{105}$.

# 中国剩余定理 (CRT)

## 中国剩余定理

方程组 $\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \vdots \\ x \equiv a_k \pmod{m_k} \end{cases}$ $(\forall i \neq j, (m_i, m_j) = 1)$ 的解

# 中国剩余定理 (CRT)

## 中国剩余定理

方程组 $\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \vdots \\ x \equiv a_k \pmod{m_k} \end{cases}$ $(\forall i \neq j, (m_i, m_j) = 1)$ 的解为

$$x \equiv \sum_{i=1}^{k} M_i' M_i a_i \pmod{M}$$

其中 $M = m_1 m_2 \cdots m_k$, $M_i = \dfrac{M}{m_i}$, $M_i' M_i \equiv 1 \pmod{m_i}$.

# 扩展中国剩余定理 (exCRT)

## 扩展中国剩余定理

求方程组 $\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \vdots \\ x \equiv a_k \pmod{m_k} \end{cases}$ 的解.

# 加法原理和乘法原理

- 加法原理: 做某件事情有几种选择, 每种选择的方案数之和就是做这件事情的方案数.

# 加法原理和乘法原理

- 加法原理: 做某件事情有几种选择, 每种选择的方案数之和就是做这件事情的方案数.
- 乘法原理: 做某件事情分为几步, 每步的方案数是独立的, 则它们的积就是做这件事情的方案数.

# 加法原理和乘法原理

- 加法原理: 做某件事情有几种选择, 每种选择的方案数之和就是做这件事情的方案数.
- 乘法原理: 做某件事情分为几步, 每步的方案数是独立的, 则它们的积就是做这件事情的方案数.

**Question.3**

求满足 $x + y \leqslant n$ 的正整数解的数量.

# 加法原理和乘法原理

- 加法原理: 做某件事情有几种选择, 每种选择的方案数之和就是做这件事情的方案数.
- 乘法原理: 做某件事情分为几步, 每步的方案数是独立的, 则它们的积就是做这件事情的方案数.

**Question.3**

求满足 $x + y \leqslant n$ 的正整数解的数量.

**Question.4**

证明: 因数个数公式: $d(n) = \prod_{i=1}^{k} (\alpha_i + 1)$.

# 排列数

## 排列

从 $n$ 个不同元素中取出 $m(m \leqslant n)$ 个元素，按照一定的顺序排成一列，叫做从 $n$ 个元素中取出 $m$ 个元素的一个排列. 所有不同的排列的个数称为排列数，记作 $P_n^m$ 或 $A_n^m$.
特别地，当 $m = n$ 时，这个排列被称作全排列.

# 排列数

## 排列

从 $n$ 个不同元素中取出 $m(m \leqslant n)$ 个元素, 按照一定的顺序排成一列, 叫做从 $n$ 个元素中取出 $m$ 个元素的一个排列. 所有不同的排列的个数称为排列数, 记作 $P_n^m$ 或 $A_n^m$.
特别地, 当 $m = n$ 时, 这个排列被称作全排列.

- 下降幂: $n^{\underline{r}} = n(n-1)(n-2)\cdots(n-r+1)$, $n^{\underline{0}} = 1$.

# 排列数

## 排列

从 $n$ 个不同元素中取出 $m(m \leqslant n)$ 个元素, 按照一定的顺序排成一列, 叫做从 $n$ 个元素中取出 $m$ 个元素的一个排列. 所有不同的排列的个数称为排列数, 记作 $P_n^m$ 或 $A_n^m$.
特别地, 当 $m = n$ 时, 这个排列被称作全排列.

- 下降幂: $n^{\underline{r}} = n(n-1)(n-2)\cdots(n-r+1)$, $n^{\underline{0}} = 1$.
- 上升幂: $n^{\overline{r}} = n(n+1)(n+2)\cdots(n+r-1)$, $n^{\overline{0}} = 1$.

# 排列数

## 排列

从 $n$ 个不同元素中取出 $m(m \leqslant n)$ 个元素, 按照一定的顺序排成一列, 叫做从 $n$ 个元素中取出 $m$ 个元素的一个排列. 所有不同的排列的个数称为排列数, 记作 $P_n^m$ 或 $A_n^m$.
特别地, 当 $m = n$ 时, 这个排列被称作全排列.

- 下降幂: $n^{\underline{r}} = n(n-1)(n-2)\cdots(n-r+1)$, $n^{\underline{0}} = 1$.
- 上升幂: $n^{\overline{r}} = n(n+1)(n+2)\cdots(n+r-1)$, $n^{\overline{0}} = 1$.
- 阶乘: $n! = n(n-1)\cdots 1$, $0! = 1$.

# 排列数

## 排列

从 $n$ 个不同元素中取出 $m(m \leqslant n)$ 个元素, 按照一定的顺序排成一列, 叫做从 $n$ 个元素中取出 $m$ 个元素的一个排列. 所有不同的排列的个数称为排列数, 记作 $P_n^m$ 或 $A_n^m$.
特别地, 当 $m = n$ 时, 这个排列被称作全排列.

- 下降幂: $n^{\underline{r}} = n(n-1)(n-2)\cdots(n-r+1)$, $n^{\underline{0}} = 1$.
- 上升幂: $n^{\overline{r}} = n(n+1)(n+2)\cdots(n+r-1)$, $n^{\overline{0}} = 1$.
- 阶乘: $n! = n(n-1)\cdots 1$, $0! = 1$.
- 排列数: $A_n^m = \dfrac{n!}{(n-m)!} = n^{\underline{m}}$.

# 组合数

## 组合

从 $n$ 个不同的元素中取出 $m(m \leqslant n)$ 个元素为一组, 叫做从 $n$ 个元素中取出 $m$ 个元素的一个组合. 所有不同的组合的个数称为组合数, 记作 $C_n^m$ 或 $\binom{n}{m}$.

# 组合数

## 组合

从 $n$ 个不同的元素中取出 $m(m \leqslant n)$ 个元素为一组, 叫做从 $n$ 个元素中取出 $m$ 个元素的一个组合. 所有不同的组合的个数称为组合数, 记作 $C_n^m$ 或 $\binom{n}{m}$.

- 组合数: $\binom{n}{m} = \frac{n^{\underline{m}}}{m!} = \frac{n!}{(n-m)!m!}$.

# 组合数

## 组合

从 $n$ 个不同的元素中取出 $m(m \leqslant n)$ 个元素为一组, 叫做从 $n$ 个元素中取出 $m$ 个元素的一个组合. 所有不同的组合的个数称为组合数, 记作 $C_n^m$ 或 $\binom{n}{m}$.

- 组合数: $\binom{n}{m} = \frac{n^{\underline{m}}}{m!} = \frac{n!}{(n-m)!m!}$.
- $\binom{n}{m} = \binom{n-1}{m-1} + \binom{n-1}{m}$. (递推求组合数).

# 组合数

## 组合

从 $n$ 个不同的元素中取出 $m(m \leqslant n)$ 个元素为一组, 叫做从 $n$ 个元素中取出 $m$ 个元素的一个组合. 所有不同的组合的个数称为组合数, 记作 $C_n^m$ 或 $\binom{n}{m}$.

- 组合数: $\binom{n}{m} = \frac{n^{\underline{m}}}{m!} = \frac{n!}{(n-m)!m!}$.
- $\binom{n}{m} = \binom{n-1}{m-1} + \binom{n-1}{m}$. (递推求组合数).
- $\binom{n}{m} = \frac{n}{m}\binom{n-1}{m-1}$.

# 组合数

## 组合

从 $n$ 个不同的元素中取出 $m(m \leqslant n)$ 个元素为一组, 叫做从 $n$ 个元素中取出 $m$ 个元素的一个组合. 所有不同的组合的个数称为组合数, 记作 $C_n^m$ 或 $\binom{n}{m}$.

- 组合数: $\binom{n}{m} = \frac{n^{\underline{m}}}{m!} = \frac{n!}{(n-m)!m!}$.
- $\binom{n}{m} = \binom{n-1}{m-1} + \binom{n-1}{m}$. (递推求组合数).
- $\binom{n}{m} = \frac{n}{m}\binom{n-1}{m-1}$.
- $\binom{n}{m} = \frac{n-m+1}{m}\binom{n}{m-1}$.

# 组合数

$$\binom{n}{m} = \binom{n-1}{m-1} + \binom{n-1}{m}, \binom{n}{0} = 1$$

| $n\backslash k$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 1 | 3 | 3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 1 | 4 | 6 | 4 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 1 | 5 | 10 | 10 | 5 | 1 | 0 | 0 | 0 | 0 | 0 |
| 6 | 1 | 6 | 15 | 20 | 15 | 6 | 1 | 0 | 0 | 0 | 0 |
| 7 | 1 | 7 | 21 | 35 | 35 | 21 | 7 | 1 | 0 | 0 | 0 |
| 8 | 1 | 8 | 28 | 56 | 70 | 56 | 28 | 8 | 1 | 0 | 0 |
| 9 | 1 | 9 | 36 | 84 | 126 | 126 | 84 | 36 | 9 | 1 | 0 |
| 10 | 1 | 10 | 45 | 120 | 210 | 252 | 210 | 120 | 45 | 10 | 1 |

# 组合数

## 不定方程解的数量

不定方程 $x_1 + x_2 + \cdots + x_k = n$ 的解的数量，其中 $x_i$ 为整数，且 $x_i \geqslant 1$.

# 组合数

## 不定方程解的数量

不定方程 $x_1 + x_2 + \cdots + x_k = n$ 的解的数量，其中 $x_i$ 为整数，且 $x_i \geqslant 1$.

- $\binom{n-1}{k-1}$.

# 组合数

## 不定方程解的数量

不定方程 $x_1 + x_2 + \cdots + x_k = n$ 的解的数量，其中 $x_i$ 为整数，且 $x_i \geqslant 1$.

- $\binom{n-1}{k-1}$.
- $x_i \geqslant a_i$ ?

# 组合数

## 不定方程解的数量

不定方程 $x_1 + x_2 + \cdots + x_k = n$ 的解的数量，其中 $x_i$ 为整数，且 $x_i \geqslant 1$.

- $\binom{n-1}{k-1}$.
- $x_i \geqslant a_i$ ?
- $x_1 + x_2 + \cdots + x_k \leqslant n$ ?

# 组合数

## 网络路径计数问题

在 $n \times m$ 的网格图上, 从 $(0,0)$ 走到 $(n,m)$, 每次只能向右走或向上走, 求方案数.

# 组合数

## 网络路径计数问题

在 $n \times m$ 的网格图上, 从 $(0,0)$ 走到 $(n,m)$, 每次只能向右走或向上走, 求方案数.

- 组合数学, $\binom{n+m}{n}$.

# 组合数

## 网络路径计数问题

在 $n \times m$ 的网格图上, 从 $(0,0)$ 走到 $(n,m)$, 每次只能向右走或向上走, 求方案数.

- 组合数学, $\binom{n+m}{n}$.
- 动态规划, `dp[i][j] = dp[i-1][j] + dp[i][j-1]`, 可以处理有障碍物的情况.

# 第二类斯特林数

## 第二类斯特林数

第二类斯特林数表示将 $n$ 个不同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $S_2(n,k)$ 或 $\left\{ {n \atop k} \right\}$.

# 第二类斯特林数

## 第二类斯特林数

第二类斯特林数表示将 $n$ 个不同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $S_2(n,k)$ 或 $\left\{ {n \atop k} \right\}$.

- 插入 1 个小球时, 有两种方案:

# 第二类斯特林数

## 第二类斯特林数

第二类斯特林数表示将 $n$ 个不同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $S_2(n,k)$ 或 $\left\{ {n \atop k} \right\}$.

- 插入 1 个小球时, 有两种方案:
  1. 将小球单独放入一个空盒子中, 有 $\left\{ {n-1 \atop k-1} \right\}$ 种方案;

# 第二类斯特林数

## 第二类斯特林数

第二类斯特林数表示将 $n$ 个不同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $S_2(n,k)$ 或 $\left\{ {n \atop k} \right\}$.

- 插入 1 个小球时, 有两种方案:
  1. 将小球单独放入一个空盒子中, 有 $\left\{ {n-1 \atop k-1} \right\}$ 种方案;
  2. 将小球放入一个现有的非空盒子中, 有 $k \left\{ {n-1 \atop k} \right\}$ 种方案.

# 第二类斯特林数

## 第二类斯特林数

第二类斯特林数表示将 $n$ 个不同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $S_2(n,k)$ 或 $\left\{ {n \atop k} \right\}$.

- 插入 1 个小球时, 有两种方案:
  1. 将小球单独放入一个空盒子中, 有 $\left\{ {n-1 \atop k-1} \right\}$ 种方案;
  2. 将小球放入一个现有的非空盒子中, 有 $k\left\{ {n-1 \atop k} \right\}$ 种方案.
- 递推式: $\left\{ {n \atop k} \right\} = \left\{ {n-1 \atop k-1} \right\} + k\left\{ {n-1 \atop k} \right\}, \left\{ {n \atop 0} \right\} = [n=0].$

# 第二类斯特林数

$$\left\{ {n \atop k} \right\} = \left\{ {n-1 \atop k-1} \right\} + k \left\{ {n-1 \atop k} \right\}, \left\{ {n \atop 0} \right\} = [n=0]$$

| $n\backslash k$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 1 | 3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 7 | 6 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 1 | 15 | 25 | 10 | 1 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 1 | 31 | 90 | 65 | 15 | 1 | 0 | 0 | 0 | 0 |
| 7 | 0 | 1 | 63 | 301 | 350 | 140 | 21 | 1 | 0 | 0 | 0 |
| 8 | 0 | 1 | 127 | 966 | 1701 | 1050 | 266 | 28 | 1 | 0 | 0 |
| 9 | 0 | 1 | 255 | 3025 | 7770 | 6951 | 2646 | 462 | 36 | 1 | 0 |
| 10 | 0 | 1 | 511 | 9330 | 34105 | 42525 | 22827 | 5880 | 750 | 45 | 1 |

# $k$ 部分拆数

## $k$ 部分拆数

$k$ 部分拆数表示将 $n$ 个相同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $p(n, k)$.

# $k$ 部分拆数

## $k$ 部分拆数

$k$ 部分拆数表示将 $n$ 个相同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $p(n, k)$.

- $k$ 部分拆数是下面方程的解的个数.

$$n - k = x_1 + x_2 + \cdots + x_k, x_1 \geq x_2 \geq \cdots \geq x_k \geq 0$$

# $k$ 部分拆数

## $k$ 部分拆数

$k$ 部分拆数表示将 $n$ 个相同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $p(n,k)$.

- $k$ 部分拆数是下面方程的解的个数.

$$n-k=x_1+x_2+\cdots+x_k, x_1\geq x_2\geq\cdots\geq x_k\geq 0$$

- 若其中有 $i$ 个数非零, 恰好有 $p(n-k,i)$ 个解.

# $k$ 部分拆数

## $k$ 部分拆数

$k$ 部分拆数表示将 $n$ 个相同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $p(n,k)$.

- $k$ 部分拆数是下面方程的解的个数.

$$n-k=x_1+x_2+\cdots+x_k, x_1\geq x_2\geq\cdots\geq x_k\geq 0$$

- 若其中有 $i$ 个数非零, 恰好有 $p(n-k,i)$ 个解.
- 可以得到 $p(n,k)=\sum_{i=0}^{k}p(n-k,i)$.

# $k$ 部分拆数

## $k$ 部分拆数

$k$ 部分拆数表示将 $n$ 个相同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $p(n,k)$.

- $p(n,k) = \sum_{i=0}^{k} p(n-k,i)$.

# $k$ 部分拆数

## $k$ 部分拆数

$k$ 部分拆数表示将 $n$ 个相同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $p(n,k)$.

- $p(n,k)=\sum_{i=0}^{k} p(n-k,i)$.
- $p(n-1,k-1)=\sum_{i=0}^{k-1} p(n-k,i)$.

# $k$ 部分拆数

## $k$ 部分拆数

$k$ 部分拆数表示将 $n$ 个相同的小球, 放入 $k$ 个相同的盒子中, 每个盒子至少放 1 个小球的不同的方案数, 记作 $p(n,k)$.

- $p(n,k)=\sum_{i=0}^{k} p(n-k,i)$.
- $p(n-1,k-1)=\sum_{i=0}^{k-1} p(n-k,i)$.
- 两式相减得递推式:
  $p(n,k)=p(n-1,k-1)+p(n-k,k), p(n,0)=[n=0]$.

# $k$ 部分拆数

$$p(n,k) = p(n-1,k-1) + p(n-k,k), p(n,0) = [n=0]$$

| $n\backslash k$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 1 | 2 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 1 | 3 | 3 | 2 | 1 | 1 | 0 | 0 | 0 | 0 |
| 7 | 0 | 1 | 3 | 4 | 3 | 2 | 1 | 1 | 0 | 0 | 0 |
| 8 | 0 | 1 | 4 | 5 | 5 | 3 | 2 | 1 | 1 | 0 | 0 |
| 9 | 0 | 1 | 4 | 7 | 6 | 5 | 3 | 2 | 1 | 1 | 0 |
| 10 | 0 | 1 | 5 | 8 | 9 | 7 | 5 | 3 | 2 | 1 | 1 |

# 球与盒子问题

## 球与盒子

$n$ 个相同/不同的小球, 放入 $k$ 个相同/不同的盒子, 每个盒子可以/不可以为空, 求方案数.

# 球与盒子问题

## 球与盒子

$n$ 个相同/不同的小球, 放入 $k$ 个相同/不同的盒子, 每个盒子可以/不可以为空, 求方案数.

| $n$ 个球 | $k$ 个盒子 | 盒子可以为空 | 盒子不可以为空 |
|---|---|---|---|
| 有标号 | 有标号 | $k^n$ | $k!S_2(n,k)$ |
| 有标号 | 无标号 | $\sum_{i=1}^{k} S_2(n,i)$ | $S_2(n,k)$ |
| 无标号 | 有标号 | $\binom{n+k-1}{k-1}$ | $\binom{n-1}{k-1}$ |
| 无标号 | 无标号 | $p(n+k,k)$ | $p(n,k)$ |

*Thanks!*