数学

计试 71 朱泽荧

2019 年 7 月 4 日

数论基础

- 约数与倍数
- 素数与合数
- 同余
- 组合数取模

整除的性质

- 性质 1: $a|b, b|c \Rightarrow a|c$
- 性质 2: $a|b \Rightarrow a|bc$
- 性质 3: $a|b, a|c \Rightarrow \forall x, y, a|xb + yc$
- 性质 4: $a|b, b|a \Leftrightarrow a = \pm b$
- 性质 5: $a = kb \pm c \Leftrightarrow a, b$ 的公因数与 b, c 的公因数完全相同（利用性质 3 证明）

最大公约数、最小公倍数

- 定义：gcd(a,b)，(a,b) 表示 a,b 的最大公约数；lcm(a,b)，[a,b] 表示 a,b 的最小公倍数

最大公约数、最小公倍数

- 定义：gcd(a,b)，(a,b) 表示 a,b 的最大公约数；lcm(a,b)，[a,b] 表示 a,b 的最小公倍数
- 基本结论
  (1) $(a,b)=(-a,b)$ ；$[a,b]=[-a,b]$；
  (2) 对任意整数 k，$(a,b)=(a,b+ka)$ ；
  (3) 设 $m>0$，则 $(am,bm)=(a,b)m$，$[am,bm]=[a,b]m$；
  (4) $(a,(b,c))=(a,b,c)$；
  (5) 若 $c|a$，$c|b$，则 $c\mid(a,b)$ ；
  (6) 若 $a|c$，$b|c$，则 $[a,b]|c$；
  (7) $|ab|=(a,b)[a,b]$。
  (8) 设 a,b,m 是整数，$(a,m)=1$，则 $(a,mb)=(a,b)$。进一步，若 $a|mb$，则 $a|b$

最大公约数、最小公倍数

- 定义：gcd(a,b)，(a,b) 表示 a,b 的最大公约数；lcm(a,b)，[a,b] 表示 a,b 的最小公倍数
- 基本结论
  (1) (a,b)=(-a,b) ； [a,b]=[-a,b]；
  (2) 对任意整数 k，(a,b)=(a,b+ka) ；
  (3) 设 m>0，则 (am,bm)=(a,b)m，[am,bm]=[a,b]m；
  (4) (a,(b,c))=(a,b,c)；
  (5) 若 c|a，c|b，则 c | (a,b) ；
  (6) 若 a|c，b|c，则 [a,b]|c；
  (7) |ab|=(a,b)[a,b]。
  (8) 设 a,b,m 是整数，(a,m)=1，则 (a,mb)=(a,b)。进一步，若 a|mb，则 a|b
- 辗转相除法原理：
  若 $a \equiv r \pmod{b}$, 则 $gcd(a,b) = gcd(b,r)$（利用性质 5 证明）

最大公约数、最小公倍数

- 定义：gcd(a,b)，(a,b) 表示 a,b 的最大公约数；lcm(a,b)，[a,b] 表示 a,b 的最小公倍数
- 基本结论
  (1) (a,b)=(-a,b) ； [a,b]=[-a,b]；
  (2) 对任意整数 k，(a,b)=(a,b+ka) ；
  (3) 设 m>0，则 (am,bm)=(a,b)m，[am,bm]=[a,b]m；
  (4) (a,(b,c))=(a,b,c)；
  (5) 若 c|a，c|b，则 c | (a,b) ；
  (6) 若 a|c，b|c，则 [a,b]|c；
  (7) |ab|=(a,b)[a,b]。
  (8) 设 a,b,m 是整数，(a,m)=1，则 (a,mb)=(a,b)。进一步，若 a|mb，则 a|b
- 辗转相除法原理：
  若 $a \equiv r \pmod{b}$, 则 $gcd(a,b) = gcd(b,r)$（利用性质 5 证明）

欧几里得算法

- 原理:
  若 $a \equiv r \pmod{b}$, 则 $gcd(a,b) = gcd(b,r)$

欧几里得算法

- 原理:
  若 $a \equiv r \pmod{b}$, 则 $gcd(a, b) = gcd(b, r)$

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

欧几里得算法

- 原理：
  若 $a \equiv r \pmod{b}$, 则 $gcd(a, b) = gcd(b, r)$
-
  ```
  int gcd(int a, int b)
  {
      return b ? gcd(b, a % b) : a;
  }
  ```
- 算法的时间复杂度：$O(log(min(a, b)))$
- 举个栗子 gcd(14,36)

欧几里得算法

- 原理:
  若 $a \equiv r \pmod{b}$, 则 $gcd(a, b) = gcd(b, r)$
-
```
int gcd(int a, int b)
{
    return b ? gcd(b, a % b) : a;
}
```
- 算法的时间复杂度: $O(log(min(a, b)))$
- 举个栗子 gcd(14,36)
- 有可能爆 int

欧几里得算法

- 原理：
  若 $a \equiv r \pmod{b}$, 则 $gcd(a,b) = gcd(b,r)$
- ```
  int gcd(int a, int b)
  {
      return b ? gcd(b, a % b) : a;
  }
  ```
- 算法的时间复杂度： $O(log(min(a,b)))$
- 举个栗子 gcd(14,36)
- 有可能爆 int
  ```

最小公倍数 LCM

求出 gcd 就可以了

lcm(a,b) = a*b / gcd(a,b)

最小公倍数 LCM

求出 gcd 就可以了

lcm(a,b) = a*b / gcd(a,b)

计算的时候最好写成 lcm(a,b) = a / gcd(a,b) * b

最小公倍数 LCM

求出 gcd 就可以了

lcm(a,b) = a*b / gcd(a,b)

计算的时候最好写成 lcm(a,b) = a / gcd(a,b) * b

为什么?

算术基本定理

素数和合数的概念

算术基本定理

素数和合数的概念

算术基本定理

任何一个大于 1 的自然数 n，都可以唯一分解成有限个质数的乘积。

$$n = p_1^{r_1} p_2^{r_2} \ldots p_k^{r_k}$$

其中 $p_1 < p_2 < \ldots < p_k$ 为质数，$r_1, r_2, \ldots, r_k$ 为正整数。

一些性质

定理

设 a 为合数，则 a 必有不超过根号 a 的素因子。

一些性质

定理

设 a 为合数，则 a 必有不超过根号 a 的素因子。

定理

若 $n=p_1^{r_1}p_2^{r_2}...p_k^{r_k}$，则 n 的约数个数为

$$(1+r_1)(1+r_2)...(1+r_k)$$

n 的所有约数和为

$$(1+p_1+p_1^2+...+p_1^{r_1})(1+p_2+p_2^2+...+p_2^{r_2})...(1+p_k+p_k^2+...+p_k^{r_k})$$

其中 $p_1<p_2<...<p_k$ 为质数，$r_1,r_2,...,r_k$ 为正整数。

POJ 1845 Sumdiv

计算 A^B 的所有约数的和模 9901
$0 \leq A, B \leq 50000000$

POJ 1845 Sumdiv

计算 A^B 的所有约数的和模 9901
$0 \leq A, B \leq 50000000$
思路：求等比数列的和

- 方法一：二分求等比数列的和

POJ 1845 Sumdiv

计算 A^B 的所有约数的和模 9901
$0 \leq A, B \leq 50000000$
思路：求等比数列的和

- 方法一：二分求等比数列的和
- 方法二：等比数列求和公式 + 逆元

Eratosthenes 筛法

求 1 到 n 中的所有素数。
每次取出第一个没被筛掉的数 p，则为素数。然后将 p 的倍数筛掉。

```
bool vis[MAXN];
int prime[MAXN];
for (int i=2;i<=n;i++)
{
    if (!vis[i])
        prime[tot++] = i;
    for (int j=i*2;j<=n;j+=i)
        vis[j] = 1;
}
```

Eratosthenes 筛法

求 1 到 n 中的所有素数。
每次取出第一个没被筛掉的数 p，则为素数。然后将 p 的倍数筛掉。

```
bool vis[MAXN];
int prime[MAXN];
for (int i=2;i<=n;i++)
{
    if (!vis[i])
        prime[tot++] = i;
    for (int j=i*2;j<=n;j+=i)
        vis[j] = 1;
}
```

时间复杂度：$n/1 + n/2 + ... + n/p = O(nlogn)$ 实际上是 $O(nloglogn)$

线性筛

一个元素会被多次筛掉。让每个元素只被它最小的质因子筛掉，可以优化成线性。

线性筛

一个元素会被多次筛掉。让每个元素只被它最小的质因子筛掉，可以优化成线性。

```
for (int i=2;i<=n;i++)
{
    if (!vis[i])
        prime[tot++] = i;
    for (int j=0;j<tot && i * prime[j] <= n;j++)
    {
        vis[i * prime[j]] = 1;
        if (i % prime[j] == 0) break;
        //保证只被最小的质因子筛掉
    }
}
```

同余的定义和性质

定义

设 m 是正整数，对整数 a,b，若 $m|a-b$，则称 a 与 b 模 m 同余，记做 $a \equiv b \pmod{m}$ 定义 $a \bmod m$ 为 0 到 $m-1$ 中和 a 同余的整数。

可以写成 $a = b + km$

同余的定义和性质

定义

设 m 是正整数，对整数 a,b，若 $m|a-b$，则称 a 与 b 模 m 同余，记做 $a \equiv b \pmod{m}$ 定义 $a \bmod m$ 为 0 到 $m-1$ 中和 a 同余的整数。

可以写成 $a = b + km$
性质:
若 $a \equiv b \pmod{m}$, $c \equiv d \pmod{m}$，则

- 性质 1: $a + c \equiv b + d \pmod{m}$
- 性质 2: $a - c \equiv b - d \pmod{m}$
- 性质 3: $a * c \equiv b * d \pmod{m}$

BZOJ 1257

- 给定正整数 n 和 k，计算 $(k \bmod 1) + (k \bmod 2) + \ldots + (k \bmod n)$ 的值，$1 \leq n, k \leq 10^9$

BZOJ 1257

- 给定正整数 n 和 k，计算 $(k \bmod 1) + (k \bmod 2) + \ldots + (k \bmod n)$ 的值，$1 \le n, k \le 10^9$
- 模数不同，考虑把求余运算变成乘法和减法。 $k \bmod i = k - \lfloor k/i \rfloor * i$，原式为 $n * k - \sum_{i=1}^{n} \lfloor k/i \rfloor * i$
- $\lfloor k/i \rfloor$ 在 $i \in [x, \lfloor k/\lfloor k/x \rfloor \rfloor]$ 内相同，在这一段中，计算一个等差数列的值

BZOJ 1257

- 给定正整数 n 和 k，计算 $(k \bmod 1) + (k \bmod 2) + ... + (k \bmod n)$ 的值，$1 \leq n, k \leq 10^9$
- 模数不同，考虑把求余运算变成乘法和减法。 $k \bmod i = k - \lfloor k/i \rfloor * i$，原式为 $n * k - \sum_{i=1}^{n} \lfloor k/i \rfloor * i$
- $\lfloor k/i \rfloor$ 在 $i \in [x, \lfloor k/\lfloor k/x \rfloor \rfloor]$ 内相同，在这一段中，计算一个等差数列的值
- 在 $i \in [1, k]$ 中，$\lfloor k/i \rfloor$ 最多只有 $2\sqrt{k}$ 个不同的值，时间复杂度 $O(\sqrt{k})$

解二元线性模方程

定义

二元线性模方程（二元一次不定方程）:
求解形如 $ax \equiv c \pmod{b}$ 或 $ax + by = c$ 的整数解

解二元线性模方程

定义

二元线性模方程（二元一次不定方程）:
求解形如 $ax \equiv c \pmod{b}$ 或 $ax + by = c$ 的整数解

- 扩展欧几里得算法与裴蜀定理

裴蜀定理

设整数 a,b 不为 0，则方程组 $ax + by = c$ 有解当且仅当 $(a, b)|c$。

解二元线性模方程

定义

二元线性模方程（二元一次不定方程）:
求解形如 $ax \equiv c \pmod{b}$ 或 $ax + by = c$ 的整数解

- 扩展欧几里得算法与裴蜀定理

裴蜀定理

设整数 a,b 不为 0，则方程组 $ax + by = c$ 有解当且仅当 $(a,b)|c$。

- 裴蜀定理特例
  若 a,b 互质，gcd(a,b)=1, 则存在 x,y 使得 $ax + by = 1$

解二元线性模方程

定义

二元线性模方程（二元一次不定方程）:
求解形如 $ax \equiv c \pmod{b}$ 或 $ax + by = c$ 的整数解

- 扩展欧几里得算法与裴蜀定理

裴蜀定理

设整数 a,b 不为 0，则方程组 $ax + by = c$ 有解当且仅当 $(a,b)|c$。

- 裴蜀定理特例
  若 a,b 互质，gcd(a,b)=1, 则存在 x,y 使得 $ax + by = 1$

因此求解 $ax + by = c$ 可以化为求解 $ax' + by' = gcd(a,b)$

推导

令 d=gcd(a,b)

$$b*x+(a\%b)*y=d \Rightarrow b*x+(a-[a/b]*b)*y=a*y+b*(x-[a/b]*y)$$

推导

令 d=gcd(a,b)

$b*x+(a\%b)*y=d \Rightarrow b*x+(a-[a/b]*b)*y=a*y+b*(x-[a/b]*y)$

因此若 $b*x+(a\%b)*y=d$ 有解 x_0, y_0，那么 $a*x+b*y=d$ 有解 $x_1=y_0, y_1=x_0-[a/b]*y_0$
可以迭代求出解

扩展欧几里得算法

求 $as + bt = gcd(a, b)$ 的解，返回 gcd(a,b)

```
int exgcd(int a, int b, int &x, int &y)
{
    if (!b) {x = 1;y = 0;return a;}
    int d = exgcd(b, a % b, y, x);
    y = y - a / b * x;//注意这里容易溢出
    return d;
}
```

扩展欧几里得算法

求 $as + bt = gcd(a, b)$ 的解，返回 gcd(a,b)

```
int exgcd(int a, int b, int &x, int &y)
{
    if (!b) {x = 1;y = 0;return a;}
    int d = exgcd(b, a % b, y, x);
    y = y - a / b * x;//注意这里容易溢出
    return d;
}
```

时间复杂度与欧几里得算法的时间复杂度相同。

通解

定理

设整数 a,b 不为 0，方程组 $ax+by=c$ 有解 x,y，则 $x+\frac{b}{(a,b)}$ 和 $y-\frac{a}{(a,b)}$ 也是一组解，且所有解都可以写成 $x+\frac{kb}{(a,b)}$ 和 $y-\frac{ka}{(a,b)}$，其中 k 是任意整数。

通解

定理

设整数 a,b 不为 0，方程组 $ax+by=c$ 有解 x,y，则 $x+\frac{b}{(a,b)}$ 和 $y-\frac{a}{(a,b)}$ 也是一组解，且所有解都可以写成 $x+\frac{kb}{(a,b)}$ 和 $y-\frac{ka}{(a,b)}$，其中 k 是任意整数。

若要求正整数解，用该定理变换。

定理

设 a,b 为正整数，则扩展欧几里得算法求出的解满足

$$|x| <= b, |y| <= a$$

证明.

由 $x_{k+1} = y_k + (a_k/b_k) * x_k, y_{k+1} = x_k$
得 $y_k = x_{k+1} - (a_k/b_k) * y_{k+1}$
下证任意时刻 $|x_k| \le b_k, |y_k| \le a_k$

定理

设 a,b 为正整数，则扩展欧几里得算法求出的解满足

$$|x| <= b, |y| <= a$$

证明.

由 $x_{k+1} = y_k + (a_k/b_k) * x_k, y_{k+1} = x_k$

得 $y_k = x_{k+1} - (a_k/b_k) * y_{k+1}$

下证任意时刻 $|x_k| \le b_k, |y_k| \le a_k$

基础：$(a_0, 0)$ 时解为 $x_0 = 1, y_0 = 0$

归纳：(a_k, b_k) 时解为 (x_k, y_k)

$(ta_k + b_k, a_k)$ 时解为 $(y_k, x_k - ty_k)$ $\square$

例

判断 $ax + by + cz = n$ 是否存在非负整数解。
数据范围： $0 \le a, b, c < 2 * 10^5, n \le 10^{18}$
Input:
1 2 3 6
3 5 6 4
Output:
YES
NO

例

判断 $ax + by + cz = n$ 是否存在非负整数解。
数据范围：$0 \le a, b, c < 2 * 10^5, n \le 10^{18}$
Input:
1 2 3 6
3 5 6 4
Output:
YES
NO
思路：若有解，则必有 $x < b$ 的解。枚举 x，用扩展欧几里得求 y,z，时间复杂度 $O(blogn)$

解线性同余方程组

定义

线性同余方程组

$$x \equiv a_1 \pmod{m_1}$$

$$\cdots$$

$$x \equiv a_n \pmod{m_n}$$

中国剩余定理 CRT

定理

当 $m_1, ..., m_n$ 两两互质时，上述线性同余方程在 $[0, m_1 m_2 ... m_n]$ 上有唯一整数解。

设同余方程组有特解 x，则所有解可表示为 $x + k m_1 m_2 ... m_n$，其中 k 是任意整数。

求解过程

令 $M = m_1 m_2 ... m_n, M_j = \frac{M}{m_j}, M_j y_j \equiv 1 \pmod{m_j}, j = 1, 2, ..., n$
因为 $gcd(M_j, m_j) = 1$，所以存在 M_j^{-1}，使 $M_j^{-1} M_j \equiv 1 \pmod{m_j}$，
且存在 h 和 k 两个整数，使得 $hM_j + km_j = 1$，$hM_j \equiv 1 \pmod{m_j}$，所以

$$y_j \equiv M_j^{-1} \pmod{m_j}$$

令 $x = M_1 y_1 a_1 + M_2 y_2 a_2 + ... + M_n y_n a_n$，易验证满足上述同余方程组。则 x 是一个解。
最小非负整数解为 $(x \bmod M + M) \bmod M$.
求解 y_j 可以用扩展欧几里得实现。
时间复杂度 $O(logm_1 + ... + logm_n)$

扩展中国剩余定理

当 $m_1, ..., m_n$ 不两两互质时，方程组不一定有解，考虑代入法解同余方程组。

扩展中国剩余定理

当 $m_1, ..., m_n$ 不两两互质时，方程组不一定有解，考虑代入法解同余方程组。
基础：a_1 是 $x \equiv a_1 \pmod{m_1}$ 的一个解
归纳：

- 设前 k-1 个方程组的一个解为 x，记 $m = lcm(m_1, m_2, ..., m_{k-1})$，则 $x + i * m$ 是前 k-1 个方程组的通解。
- 对第 k 个方程，求出一个 t 使得

$$x + t * m \equiv a_k \pmod{m_k}$$

- 用扩展欧几里得解 t，并可以判断有没有解。

扩展中国剩余定理

当 $m_1, \ldots, m_n$ 不两两互质时，方程组不一定有解，考虑代入法解同余方程组。
基础：a_1 是 $x \equiv a_1 \pmod{m_1}$ 的一个解
归纳：

- 设前 k-1 个方程组的一个解为 x，记 $m = lcm(m_1, m_2, \ldots, m_{k-1})$，则 $x + i * m$ 是前 k-1 个方程组的通解。
- 对第 k 个方程，求出一个 t 使得

$$x + t * m \equiv a_k \pmod{m_k}$$

- 用扩展欧几里得解 t，并可以判断有没有解。

为什么模数变成了 lcm?

代入过程

假设要合并 $x \equiv a_1 \pmod m_1$

$x \equiv a_2 \pmod m_2$

就需要求出一个 x 满足 $x = a_1 + m_1 * k_1 = a_2 + m_2 * k_2$

代入过程

假设要合并 $x \equiv a_1 \pmod m)_1$
$x \equiv a_2 \pmod m)_2$
就需要求出一个 x 满足 $x = a_1 + m_1 * k_1 = a_2 + m_2 * k_2$

$$m_1 * k_1 - m_2 * k_2 = a_2 - a_1$$

可由扩展欧几里得算法求出一组解 x_1, y_1，满足
$m_1 * x_1 + m_2 * y_1 = gcd(m_1, m_2)$

代入过程

假设要合并 $x \equiv a_1 \pmod m_1$
$x \equiv a_2 \pmod m_2$
就需要求出一个 x 满足 $x = a_1 + m_1 * k_1 = a_2 + m_2 * k_2$

$$m_1 * k_1 - m_2 * k_2 = a_2 - a_1$$

可由扩展欧几里得算法求出一组解 x_1, y_1，满足
$m_1 * x_1 + m_2 * y_1 = gcd(m_1, m_2)$ 在 $gcd(m_1, m_2)|(a_2 - a_1)$ 时，有
$m_1 * x_1 * \frac{a_2 - a_1}{gcd(m_1, m_2)} + m_2 * x_2 * \frac{a_2 - a_1}{gcd(m_1, m_2)} = a_2 - a_1$ （否则无解）
则 $k_1^* = x_1 * \frac{a_2 - a_1}{gcd(m_1, m_2)}$

代入过程

假设要合并 $x \equiv a_1 \pmod m_1$
$x \equiv a_2 \pmod m_2$
就需要求出一个 x 满足 $x = a_1 + m_1 * k_1 = a_2 + m_2 * k_2$

$$m_1 * k_1 - m_2 * k_2 = a_2 - a_1$$

可由扩展欧几里得算法求出一组解 x_1, y_1，满足
$m_1 * x_1 + m_2 * y_1 = gcd(m_1, m_2)$ 在 $gcd(m_1, m_2)|(a_2 - a_1)$ 时，有
$m_1 * x_1 * \frac{a_2 - a_1}{gcd(m_1, m_2)} + m_2 * x_2 * \frac{a_2 - a_1}{gcd(m_1, m_2)} = a_2 - a_1$ （否则无解）
则 $k_1^* = x_1 * \frac{a_2 - a_1}{gcd(m_1, m_2)}$
通解 $k_1 = k_1^* + m_2/gcd(m_1, m_2) * T$ 可以找到最小的正整数解 k_1，并代入 $x = a_1 + k_1 * m_1$ 得到

$$x = a_1 + (k_1^* + T * m_2/gcd(m_1, m_2)) * m_1$$

所以合并后的方程变成 $x \equiv A \pmod{lcm(m_1, m_2)}$

快速乘

为了解决上面过程中乘法爆 long long 的问题

```
typedef long long LL;
LL qmul(LL n, LL b, LL P) // a * b % P
{
    LL ans = 0;
    while (b)
    {
        if (b & 1) ans = (ans + a) % P;
        b >>= 1;
        a = (a + a) % P;
    }
    return ans;
}
```

hdu 1573 X 问题

题目描述：
求在小于等于 N 的正整数中有多少个 X 满足：$X \bmod a[0] = b[0]$, $X \bmod a[1] = b[1]$, $X \bmod a[2] = b[2]$, ..., $X \bmod a[i] = b[i]$, ...$(0 < a[i] <= 10)$。
输入：N,M，数组 a,b
例子：
Input
10 3
1 2 3
0 1 2
Output
1

hdu1573 X 问题 Solution

题目没有保证 a[i] 互质，用扩展中国剩余定理。

定理

若 $ac \equiv bc \pmod{m}$, 则 $a \equiv b \pmod{\frac{m}{gcd(m,c)}}$。进一步，若 $gcd(c, m) = 1$，则 $a \equiv b \pmod{m}$

定理

若 $a \equiv b \pmod{m_i}$, $i = 1, 2, ..., n$，则 $a \equiv b \pmod{lcm(m_1, m_2, ..., m_n)}$

欧拉函数

定义

欧拉函数 $\varphi(n)$ 为 1 到 n 中与 n 互质的数的个数。

性质：

- 积性：(m,n)=1 时，$\varphi(mn)=\varphi(m)\varphi(n)$
- 设 p 是素数，$\varphi(p)=p-1$
- 设 p 是素数，$k>1$，则 $\varphi(p^k)=p\varphi(p^{k-1})$
- 设 p 是素数，且 p | n，则 $\varphi(pn)=p\varphi(n)$

欧拉函数计算公式：$\varphi(n)=n(1-\frac{1}{p_1})(1-\frac{1}{p_2})...(1-\frac{1}{p_k})$（容斥原理）

欧拉公式：设 a 与 m 互质，则

$$a^{\varphi(m)} \equiv 1 \pmod{m}$$

欧拉函数的性质

- 设 $(a,m)=1$，则 $a^n \equiv a^{n \bmod \varphi(m)} \pmod m$
- 设 n 是满足 $a^n \equiv 1 \pmod m$ 的最小正整数，则有 $n|\varphi(m)$。n 记做 a 关于模 m 的阶。进一步，若 $a^x \equiv 1 \pmod m$, $a^y \equiv 1 \pmod m$，则 $a^{(x,y)} \equiv 1 \pmod m$。
- 设 $n>1$，则所有小于 n 且与 n 互质的数的和为 $n\varphi(n)/2$。

例

求第 k 大的 $\varphi(n)$ 为合数的数 n.

例

求第 k 大的 $\varphi(n)$ 为合数的数 n.

$\varphi(1) = 1$

$\varphi(2) = 1$

$\varphi(3) = 2$

$\varphi(4) = 2$

$\varphi(5) = 4$ ✓

$\varphi(6) = 2$

$\varphi(7) = 6$ ✓

...

例

求第 k 大的 $\varphi(n)$ 为合数的数 n.

$\varphi(1) = 1$

$\varphi(2) = 1$

$\varphi(3) = 2$

$\varphi(4) = 2$

$\varphi(5) = 4$ ✓

$\varphi(6) = 2$

$\varphi(7) = 6$ ✓

...

$$\varphi(n) = n(1 - \frac{1}{p_1})(1 - \frac{1}{p_2})...(1 - \frac{1}{p_k})$$

例

求第 k 大的 $\varphi(n)$ 为合数的数 n.

$\varphi(1) = 1$
$\varphi(2) = 1$
$\varphi(3) = 2$
$\varphi(4) = 2$
$\varphi(5) = 4$ ✓
$\varphi(6) = 2$
$\varphi(7) = 6$ ✓
...

$$\varphi(n) = n(1 - \frac{1}{p_1})(1 - \frac{1}{p_2})...(1 - \frac{1}{p_k})$$

k=1 时，答案是 5; k>1 时，答案是 k+5
事实上，

定理

若 $n > 2$，$\varphi(n)$ 必定是偶数。

求欧拉函数

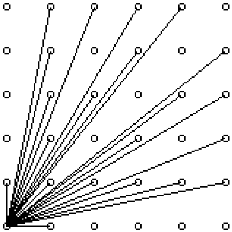
- 求 1 到 n 的所有 $\varphi(i)$：采用欧拉筛的思想，时间复杂度 $O(n)$。
- 求 $\varphi(n)$：用分解质因数的方法，时间复杂度 $O(\sqrt{n})$

筛法求欧拉函数

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

洛谷 2158 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 * 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)。现在，C 君希望你告诉他队伍整齐时能看到的学生人数。



洛谷 2158 仪仗队题解

- 能看见的点关于 $y = x$ 对称
- $ANS(1) = 0$
- $ANS(n) = \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} [gcd(x, y) == 1], n \geq 2$
- $= 2\sum_{x=1}^{n-1} \varphi(x) + 1$

欧拉定理推广

欧拉定理

若 a 和 n 互质，则 $a^{\varphi(m)} \equiv 1 \pmod{m}$

费马小定理

设 p 是素数，$(a, p) = 1$，则 $a^{p-1} \equiv 1 \pmod{p}$

欧拉定理推广

欧拉定理

若 a 和 n 互质，则 $a^{\varphi(m)} \equiv 1 \pmod{m}$

费马小定理

设 p 是素数，$(a, p) = 1$，则 $a^{p-1} \equiv 1 \pmod{p}$

推广 (降幂公式)

当 $x \geq \varphi(m)$ 时，$a^x \equiv a^{x \bmod \varphi(m) + \varphi(m)} \pmod{m}$

例

计算

$$\underbrace{a^{a^{a^{\cdots}}}}_{k\text{个}a} \bmod 100000000$$

$1 \le k \le 200, 1 \le a \le 10^{18}$

例

计算

$$\underbrace{a^{a^{a^{\cdots}}}}_{k\text{个}a} \bmod 100000000$$

$1 \le k \le 200, 1 \le a \le 10^{18}$
思路：按照题意递归计算
$a^x \equiv a^{x \bmod \varphi(n) + \varphi(n)} \pmod n$
当指数比欧拉函数大的时候，根据降幂公式可以取模
C = 1e8，预处理出需要用到的欧拉函数：$C, \varphi(C), \varphi(\varphi(C)), \ldots$
(200 个)

同余类与剩余系

定义

对于 $\forall a \in [0, m-1]$，集合 $\{a+km\}(k \in \mathbb{Z})$ 的所有数模 m 同余，余数都是 a。该集合成为一个模 m 的同余类，记为 $\bar{a}$. 模 m 的同余类有 m 个，分别为 $\overline{0}, \overline{1}, ..., \overline{m-1}$，它们构成 m 的完全剩余系。$1 \sim m$ 中与 m 互质的数代表的同余类有 $\varphi(m)$ 个，它们构成 m 的简化剩余系。

群

群 $(S, +)$ 是一个集合 S 和定义在 S 上的二元运算 +，它满足如下性质：

- 封闭性：如果 $a, b \in S$，那么 $a + b \in S$
- 单位元：存在一个元素 e，使得对于所有 $a \in S$ 都满足 $e + a = a + e = a$
- 结合律：对于任意 a,b,c 都满足 (a+b)+c=a+(b+c)
- 逆元：对每个 $a \in S$ 都存在唯一的元素 $b \in S$ 使得 a+b=b+a=e。把 b 称作 a 的逆元。

根据模加法和模乘法定义的群：

- 定义在集合 $Z_n = \{0, 1, ..., n-1\}$ 上
- 集合上的加法和乘法运算定义为：$[a]_n +_n [b]_n = [a+b]_n$ $[a]_n *_n [b]_n = [a*b]_n$

逆元

定义

设 m,a 为正整数，若存在正整数 a'，满足 $aa' \equiv 1 \pmod{m}$，则称 a' 是 a 关于模 m 的逆元，记为 a^{-1}.

一些定理:

1. a 的逆元存在当且仅当 $(a, m) = 1$；且逆元唯一。
2. 同余方程 $ax \equiv b \pmod{m}$ 当 $(a, m) = 1$ 时在区间 $0 \leq x < m$ 上有唯一解 $a^{-1}b$
3. 同余方程 $ax \equiv b \pmod{m}$ 有解当且仅当 $(a, m)|b$，且在区间 $0 \leq x < m/(a, m)$ 上有唯一解，在区间 $0 \leq x < m$ 上有 (a, m) 个解。

定理

同余方程 $ax \equiv b \pmod{m}$ 当 $(a,m)=1$ 时在区间 $0 \leq x < m$ 上有唯一解 $a^{-1}b$

证明.

存在性：由裴蜀定理可知，存在 x,y 满足 $ax+my=(a,m)=1$，则存在整数 b 使得 $x'=bx, y'=by$ 满足 $ax'+my'=b$，即 $ax \equiv b \pmod{m}$ 有解 bx。通解为 $bx+km$，则必在 $[0,m-1]$ 上有一个解。

唯一性：假设在 $[0,m-1]$ 上有两个解 x_1, x_2 满足 $x_1 \geq x_2$，则 $ax_1 \equiv b \pmod{m}$，$ax_2 \equiv b \pmod{m}$。则 $a(x_1-x_2) \equiv 0 \pmod{m}$，且 $x_1-x_2 \in [0,m-1]$，所以 $x_1=x_2$ $\square$

定理

同余方程 $ax \equiv b \pmod m$ 有解当且仅当 $(a,m)|b$，且在区间 $0 \le x < m/(a,m)$ 上有唯一解，在区间 $0 \le x < m$ 上有 (a,m) 个解。

证明.

令 $d = gcd(a,m)$，$ax \equiv b \pmod m$，则 $ax + my = b$

$$\frac{a}{d}x + \frac{m}{d}y = \frac{b}{d}$$

$gcd(\frac{a}{d}, \frac{m}{d}) = 1$, 由定理 2，该方程在 $[0, \frac{m}{d} - 1]$ 上有唯一解。
若有特解 x，则所有解为 $\{x + k\frac{m}{d}, k \in \mathbb{Z}\}$，在模 m 的完全剩余系 $\{\overline{0}, \overline{1}, ..., \overline{m-1}\}$ 中，恰有 (a,m) 个解。 □

由此可以解释关于拓展欧几里得算法通解的定理。

求逆元

- 方法一：费马小定理
  设 p 是素数，且 gcd(a,p)=1，则 $a^{p-1} \equiv 1(mod p)$
  当 m 为素数时，a^{m-2} 是 a 关于模 m 的逆元。（快速幂）
  当 m 不是素数时，$a^{\varphi(m)-1}$ 是 a 关于模 m 的逆元。

求逆元

- 方法一：费马小定理
  设 p 是素数，且 gcd(a,p)=1，则 $a^{p-1} \equiv 1(modep)$
  当 m 为素数时，a^{m-2} 是 a 关于模 m 的逆元。（快速幂）
  当 m 不是素数时，$a^{\varphi(m)-1}$ 是 a 关于模 m 的逆元。
- 方法二：扩展欧几里得
  对 m 是否为素数没有限制。
  使用扩展欧几里得求 $ax \equiv 1(modm)$
  （化为 $ax + my = 1$）

求逆元

- 方法一：费马小定理
  设 p 是素数，且 gcd(a,p)=1，则 $a^{p-1} \equiv 1(mod p)$
  当 m 为素数时，a^{m-2} 是 a 关于模 m 的逆元。(快速幂)
  当 m 不是素数时，$a^{\varphi(m)-1}$ 是 a 关于模 m 的逆元。
- 方法二：扩展欧几里得
  对 m 是否为素数没有限制。
  使用扩展欧几里得求 $ax \equiv 1(mod m)$
  (化为 $ax + my = 1$)
- 两种方法的时间复杂度均为 $O(log m)$

线性时间求 1 到 n 中所有数模素数 m 的逆元

```
typedef long long LL;
int inv[N], m;
void get_inv(int n)
{
    inv[1] = 1;
    for (int i=2;i<=n;i++)
        inv[i] = (LL)(m-m/i) * inv[m % i] % m;
}
```

证明：设 $m = ki + b$，则 $ki \equiv -b \pmod{m}$，

$i^{-1} \equiv -b^{-1}k \equiv -b^{-1}(m-k) \pmod{m}$

即 $inv[i] = (LL)(m - m/i) * inv[m\%i]\%m$;

阶乘的逆元

求 1 到 n 的阶乘的逆元

```
fac[0] = 1;
for (int i=1;i<=n;i++)
    fac[i] = fac[i-1] * i % P;
invfac[n] = qpow(fac[n], P-2); //快速幂
for (int i=n;i>=1;i--)
    invfac[i-1] = invfac[i] * i % P;
```

可以用来求单个排列组合数取模

组合数取模

- 求 $C_n^m \mod p$
- 根据 n,m,p 的范围和约束条件不同，求解方法不同。

n,m 较小，p 较大

$1 \le m \le n \le 1000, 1 \le p \le 10^9$

$$C_n^m = C_{n-1}^{m-1} + C_{n-1}^m$$

直接利用杨辉三角，递推计算组合数，时间复杂度 $O(n^2)$

n,m 较大，p 较小且 p 为素数

$1 \leq m \leq n \leq 10^{18}, 2 \leq p \leq 10^5$ 且 p 是素数

n,m 较大，p 较小且 p 为素数

$1 \leq m \leq n \leq 10^{18}, 2 \leq p \leq 10^5$ 且 p 是素数
直接计算显然不可能

n,m 较大，p 较小且 p 为素数

$1 \le m \le n \le 10^{18}, 2 \le p \le 10^5$ 且 p 是素数
直接计算显然不可能

Lucas 定理

如果 p 是素数，对整数 $1 \le m \le n$ 有，

$$C_n^m \equiv C_{n \bmod p}^{m \bmod p} * C_{n/p}^{m/p} \pmod{p}$$

也就是将 n 和 m 分别表示成 p 进制数，对 p 进制下的每一位计算组合数。
对 $C_{n \bmod p}^{m \bmod p}$ 的计算，用组合数阶乘的公式和逆元。
递归计算，出口为 m=0
时间复杂度 $O(plog_p^n)$

n,m 较大，p 较小但 p 可能为合数

扩展 Lucas

思路：考虑将 p 分解，得到同余方程组，再用中国剩余定理合并。

n,m 较大，p 较小但 p 可能为合数

扩展 Lucas

思路：考虑将 p 分解，得到同余方程组，再用中国剩余定理合并。

- 令 $p = p_1^{r_1} p_2^{r_2} ... p_k^{r_k}$，需要求出 $C_n^m \equiv c_i \pmod{p_i^{r_i}}, i=1,...k$

n,m 较大，p 较小但 p 可能为合数

扩展 Lucas

思路：考虑将 p 分解，得到同余方程组，再用中国剩余定理合并。

- 令 $p = p_1^{r_1} p_2^{r_2} ... p_k^{r_k}$，需要求出 $C_n^m \equiv c_i \pmod{p_i^{r_i}}, i=1,...k$
-

$$C_n^m \equiv \frac{n!}{m!(n-m)!}$$

n,m 较大，p 较小但 p 可能为合数

扩展 Lucas

思路：考虑将 p 分解，得到同余方程组，再用中国剩余定理合并。

- 令 $p = p_1^{r_1} p_2^{r_2} \ldots p_k^{r_k}$，需要求出 $C_n^m \equiv c_i \pmod{p_i^{r_i}}$,$i$=1,...$k$
- $$C_n^m \equiv \frac{n!}{m!(n-m)!} \equiv \frac{\frac{n!}{p^{a_1}}}{\frac{m!}{p^{a_2}} * \frac{(n-m)!}{p^{a_3}}} * p^{a_1-a_2-a_3} \pmod{p^r}$$

  其中 a_1, a_2, a_3 分别是 $[n/p], [m/p], [(n-m)/p]$
  不能直接求逆元，因为分母与模数不互质，只要把公因子提出来就好了，提出来之后就可以直接求逆元

n,m 较大，p 较小但 p 可能为合数

扩展 Lucas

思路：考虑将 p 分解，得到同余方程组，再用中国剩余定理合并。

- 令 $p = p_1^{r_1} p_2^{r_2} \ldots p_k^{r_k}$，需要求出 $C_n^m \equiv c_i \pmod{p_i^{r_i}}, i=1,\ldots k$
- $$C_n^m \equiv \frac{n!}{m!(n-m)!} \equiv \frac{\frac{n!}{p^{a_1}}}{\frac{m!}{p^{a_2}} * \frac{(n-m)!}{p^{a_3}}} * p^{a_1-a_2-a_3} \pmod{p^r}$$

  其中 a_1, a_2, a_3 分别是 $[n/p], [m/p], [(n-m)/p]$
  不能直接求逆元，因为分母与模数不互质，只要把公因子提出来就好了，提出来之后就可以直接求逆元

扩展 Lucas 续

- 考虑计算

$$\frac{n!}{p^{[n/p]}} \bmod p^k$$

先看 $n! \bmod p^k$，
比如 n=19, p=3, k=2 时
$19! = 1 * 2 * 3 * ... * 19$

扩展 Lucas 续

- 考虑计算

$$\frac{n!}{p^{[n/p]}} \bmod p^k$$

先看 $n! \bmod p^k$,
比如 n=19, p=3, k=2 时
$19! = 1 * 2 * 3 * ... * 19 = (1 * 2 * 4 * 5 * 7 * 8 * 10 * 11 * 13 * 14 * 16 * 17 * 19) * 3^6 * 6!$

扩展 Lucas 续

- 考虑计算

$$\frac{n!}{p^{[n/p]}} \mod p^k$$

先看 $n! \mod p^k$，
比如 n=19, p=3, k=2 时
$19! = 1*2*3*...*19 = (1*2*4*5*7*8*10*11*13*14*16*17*19)*3^6*6! = (1*2*4*5*7*8)^2*19*3^6*6!$
因为 $1*2*4*5*7*8 \equiv 10*11*13*14*16*17 \mod 3^2$
即 $\prod_{i,(i,p)=1}^{p^k} i \equiv \prod_{i,(i,p)=1}^{p^k}(i+t*p^k) \pmod{p^k}$

$$n! \equiv p^{[\frac{n}{p}]} * [\frac{n}{p}]! * (\prod_{i,(i,p)=1}^{p^k} i)^{\frac{n}{p^k}} * (\prod_{i,(i,p)=1}^{n \mod p^k} i) \pmod{p^k}$$

谢谢！