简单初等数论入门

真的很简单

前言

- 由于时间比较紧，只会对每个知识点进行略讲；
- 我会提供足够的习题，请同学们课后自行练习以保证效果；
- **我是小萌新题目不难大家轻喷**；
- 想要深入学习推荐 OI-Wiki 的数论板块；

默认大家都会的幼儿园芝士

- 整除、约数（因数）的定义
- 带余数除法（取模）
- 最大公约数与最小公倍数的定义
- 素数（质数）与合数的定义
- 算术基本定理（唯一分解定理）
- 同余及其基本性质
- 素数筛法

欧几里得算法

用于求最大公约数。

$$\gcd(a, b) = \gcd(b, a \bmod b)$$

手动实现:

```
int gcd(int a, int b)
{
    while (b ≠ 0)
    {
        int tmp = a;
        a = b; b = tmp % b;
    }
    return a;
}
```

C++ 14 可用函数:

```
__gcd(a,b);
```

裴蜀定理

- 定义：关于 x, y 的二元一次不定方程 $ax + by = c$ 有整数解的充要条件是 $\gcd(a, b) \mid c$。
- 推广：裴蜀定理可以推广到 n 个整数的情形。

例题：P4549

给定一个包含 n 个元素的**整数**序列 A，记作 $A_1, A_2, A_3, ..., A_n$。求另一个包含 n 个元素的待定**整数**序列 X，记 $S = \sum_{i=1}^{n} A_i \times X_i$，使得 $S > 0$ 且 S 尽可能的小。$1 \le n \le 20$，$|A_i| \le 10^5$，且 A 序列不全为 0。

答案即为 $\gcd(A_1, A_2, A_3, \cdots, A_n)$

扩展欧几里得算法（exgcd）

常用于求 $ax+by=\gcd(a,b)$ 的一组可行解。

过程

首先，当 a, b 不全为 0，$ax + by = \gcd(a, b)$ 一定有解，即裴蜀定理。

当 b 等于 0 时，$\gcd(a, b) = a$，此时 $x = 1, y = 0$，显然是方程的一组特解。

而当 b 不等于 0 时，由 $\gcd(a, b) = \gcd(b, a \bmod b)$ 可以知道：$ax + by = bx' + (a \bmod b)y'$，而：

$$\begin{aligned} bx' + (a \bmod b)y' &= bx' + (a - \lfloor \frac{a}{b} \rfloor \times b)y' \\ &= ay' + bx' - \lfloor \frac{a}{b} \rfloor \times by' \\ &= ay' + b(x' - \lfloor \frac{a}{b} \rfloor y') \end{aligned}$$

由 $ax + by = ay' + b(x' - \lfloor \frac{a}{b} \rfloor y')$ 可以得到：$x = y', y = x' - \lfloor \frac{a}{b} \rfloor y'$。

将 x', y' 不断代入递归求解直至 gcd 为 0 再递归 $x = 1, y = 0$ 回去求解即可。

实现

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

扩展欧几里得算法

例题：P5656

给定不定方程 $ax+by=c$。若该方程无整数解，输出 -1。
若该方程有整数解，且有正整数解，则输出其**正整数**解的数量，所有**正整数**解中 x 的最小值，所有**正整数**解中 y 的最小值，所有**正整数**解中 x 的最大值，以及所有**正整数**解中 y 的最大值。
若方程有整数解，但没有正整数解，你需要输出所有**整数解**中 x 的最小正整数值，y 的最小正整数值。

对于形如 $ax+by=c$ 的二元一次方程，显然当且仅当 $\gcd(a,b)\mid c$ 时，存在整数解。

设 $d=\gcd(a,b), a'=a/d, b'=b/d, c'=c/d$，此时 $\gcd(a',b')=1$，我们只需要求出 $a'x'+b'y'=1$ 的一组特解 x'_0,y'_0，原方程的其中一组解便是 $x_0=x'_0\times c, y_0=y'_0\times c$。

求出特解后，该方程的通解便是：$x=x_0+b't, y=y_0-a't$，其中 t 是任意整数。

费马小定理

定义有两种等价的形式:

- 若 p 为质数，$\gcd(a,p)=1$，则 $a^{p-1}\equiv 1(\mathrm{mod} p)$
- 若 p 为质数，对于任意整数 a，$a^p\equiv a(\mathrm{mod} p)$

证明

显然 $1^p\equiv 1(\mathrm{mod} p)$ 成立，我们假设 $a^p\equiv a(\mathrm{mod} p)$ 成立，那么:

$$(a+1)^p=a^p+\binom{p}{1}a^{p-1}+\binom{p}{2}a^{p-2}+\cdots+\binom{p}{p-1}a+1$$

而由组合数的定义，我们知道 $\forall 1\leq k\leq p-1, \binom{p}{k}\equiv 0(\mathrm{mod} p)$ 。

那么 $(a+1)^p\equiv a^p+1(\mathrm{mod} p)$，将 $a^p\equiv a(\mathrm{mod} p)$ 代入即可得到 $(a+1)^p\equiv a+1(\mathrm{mod} p)$。

费马小定理

另一种证明

考虑 $p-1$ 个整数 $a, 2a, 3a, \cdots, (p-1)a$，其中 $\gcd(a,p)=1$。易知它们每个数模 p 都是独一无二的，证明如下：

若存在两个数在模 p 意义下相同，设它们分别为 xa 和 ya（$1 \le x, y < p$），即：$xa \equiv ya (\bmod p)$。两边同时消去 a 得 $x \equiv y (\bmod p)$，矛盾。由此可以得到：

$$a \times 2a \times \cdots \times (p-1)a \equiv 1 \times 2 \times \cdots \times (p-1) (\bmod p)$$

即：

$$a^{p-1} \times (p-1)! \equiv (p-1)! (\bmod p)$$

两边同时消去 $(p-1)!$ 得：

$$a^{p-1} \equiv 1 (\bmod p)$$

欧拉函数

表示小于等于 n 和 n 互质的数的个数。写作 $\varphi(n)$。

比如：$\varphi(1) = 1, \varphi(7) = 6, \varphi(12) = 4$

欧拉函数的性质

- 当 n 是质数时，$\varphi(n) = n - 1$
- 欧拉函数是积性函数，即若有 $\gcd(a, b) = 1$，$\varphi(a \times b) = \varphi(a) \times \varphi(b)$。
- $n = \sum_{d|n} \varphi(d)$。

欧拉函数的性质

- 由唯一分解定理，设 $n=\prod_{i=1}^{s} p_i^{k_i}$，其中 p_i 是质数，有 $\varphi(n)=n\times\prod_{i=1}^{s}\frac{p_i-1}{p_i}$

证明

$$
\begin{aligned}
\varphi(n) &= \prod_{i=1}^{s}\varphi(p_i^{k_i}) \\
&= \prod_{i=1}^{s}(p_i-1)\times p_i^{k_i-1} \\
&= \prod_{i=1}^{s}p_i^{k_i}\times(1-\frac{1}{p_i}) \\
&= n\prod_{i=1}^{s}(1-\frac{1}{p_i})
\end{aligned}
$$

线性求欧拉函数

注意到在线性筛中，每一个合数都是被最小的质因子筛掉。比如设 p_1 是 n 的最小质因子，$n' = \frac{n}{p_1}$，那么线性筛的过程中 n 通过 $n' \times p_1$ 筛掉。

```
vector<int> pri;
bool not_prime[N];
void pre(int n)
{
    for (int i = 2; i <= n; ++i)
    {
        if (!not_prime[i]) pri.push_back(i);
        for (int pri_j : pri)
        {
            if (i * pri_j > n) break;
            not_prime[i * pri_j] = true;
            if (i % pri_j == 0) break;
        }
    }
}
```

线性求欧拉函数

观察线性筛的过程，我们还需要处理两个部分，下面对 $n' \bmod p_1$ 分情况讨论。

如果 $n' \bmod p_1 = 0$，那么 n' 包含了 n 的所有质因子。

$$\begin{aligned}\varphi(n) &= n \times \prod_{i=1}^{s} \frac{p_i - 1}{p_i} \\ &= p_1 \times n' \times \prod_{i=1}^{s} \frac{p_i - 1}{p_i} \\ &= p_1 \times \varphi(n')\end{aligned}$$

那如果 $n' \bmod p_1 \neq 0$ 呢，这时 n' 和 p_1 是互质的，根据欧拉函数性质，我们有：

$$\begin{aligned}\varphi(n) &= \varphi(p_1) \times \varphi(n') \\ &= (p_1 - 1) \times \varphi(n')\end{aligned}$$

```cpp
vector<int> pri;
bool not_prime[N];
int phi[N];
void pre(int n)
{
    phi[1] = 1;
    for (int i = 2; i <= n; i++)
    {
        if (!not_prime[i])
        {
            pri.push_back(i);
            phi[i] = i - 1;
        }
        for (int pri_j : pri)
        {
            if (i * pri_j > n) break;
            not_prime[i * pri_j] = true;
            if (i % pri_j == 0)
            {
                phi[i * pri_j] = phi[i] * pri_j;
                break;
            }
            phi[i * pri_j] = phi[i] * phi[pri_j];
        }
    }
```

欧拉定理

- 若 $\gcd(a,m)=1$，则 $a^{\varphi(m)}\equiv 1(\bmod m)$

证明

与费马小定理的第二种证明非常相似。

设小于 m 且与 m 互质得数组成 $r_1,r_2,\cdots,r_{\varphi(m)}$。仍然考虑 $\varphi(m)$ 个数 $r_1a,r_2a,\cdots,r_{\varphi(m)}a$，其中 $\gcd(a,m)=1$，也很容易知道这 $\varphi(m)$ 个数两两不同，且由于 $\gcd(r_ia,m)$ 互质，可以得到 $r_1,r_2,\cdots,r_{\varphi(m)}$ 与 $r_1a,r_2a,\cdots,r_{\varphi(m)}a$ 中的数一一对应。所以：

$$r_1\times r_2\times\cdots\times r_{\varphi(m)}\equiv r_1a\times r_2a\times\cdots\times r_{\varphi(m)}a\equiv a^{\varphi(m)}\times r_1\times r_2\times\cdots\times r_{\varphi(m)}(\bmod m)$$

同时约去 $r_1\times r_2\times\cdots\times r_{\varphi(m)}$，即得：

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

可以发现，费马小定理其实是欧拉定理在 m 为质数的一种特殊情况。

扩展欧拉定理

$$a^b \equiv \begin{cases} a^b, b < \varphi(m) \\ a^{b \bmod \varphi(m) + \varphi(m)}, b \geq \varphi(m) \end{cases} \pmod{m}$$

证明：https://oi-wiki.org/math/number-theory/fermat/#证明_2

例题：P5091

给你三个正整数，a, m, b，你需要求：$a^b \bmod m$

对于 100% 的数据，$1 \leq a \leq 10^9$，$1 \leq b \leq 10^{20000000}$，$1 \leq m \leq 10^8$。

预处理出 $\varphi(m)$，用快读边读入边取模即可。

模意义下的乘法逆元

- 如果一个线性同余方程 $ax \equiv 1 (\bmod b)$，则 x 被称作 $a \bmod b$ 的逆元，记作 a^{-1}
- 乘法逆元常用来解决有理数取模的问题。

例题：P2613

给出一个有理数 $c = \frac{a}{b}$，求 $c \bmod 19260817$ 的值。

即求 $a \times b^{-1} \bmod 19260817$。

求单个数的逆元

扩展欧几里得算法

$ax \equiv 1 (\bmod b)$ 与 $ax + by = 1$ 是等价的，因此就是求 x 的最小正整数解。

注意需满足 $\gcd(a, b) = 1$

快速幂法

$x \equiv a^{b-2} (\bmod b)$ ，用快速幂求即可。证明如下:

因为 $ax \equiv 1 (\bmod b)$，根据费马小定理，$ax \equiv a^{b-1} (\bmod b)$，所以 $x \equiv a^{b-2} (\bmod b)$

注意需满足 b 是质数。

线性求逆元

例题：P3811

线性复杂度求出 $1, 2, \cdots, n$ 中每个数关于 p 的逆元。

线性求逆元

首先，1 的逆元是它本身，即 $1^{-1} \equiv 1(\bmod p)$

对于递归情况 i^{-1}，我们设 $k = \lfloor \frac{p}{i} \rfloor, j = p \bmod i$，有 $p = ki + j$，那么：

$$ki + j \equiv 0(\bmod p)$$

在两边同时乘 $i^{-1} \times j^{-1}$：

$$\begin{aligned} kj^{-1} + i^{-1} &\equiv 0(\bmod p) \\ i^{-1} &\equiv -kj^{-1}(\bmod p) \\ i^{-1} &\equiv -\lfloor \frac{p}{i} \rfloor (p \bmod i)^{-1}(\bmod p) \end{aligned}$$

而 $p \bmod i < i$，因此我们可以从已经计算出的答案得到 i^{-1}。

```
inv[1]=1;
for(int i=2;i<=n;i++)
    inv[i]=1ll*(p-p/i)*inv[p%i]%p;
```

线性求任意 n 个数的逆元

例题：P5431

线性复杂度求出任意给定 n 个数（$1 \leq a_i < p$）的逆元。

首先计算 n 个数的前缀积，记为 s_i，计算出 s_n 的逆元，记为 sv_n。

由于 sv_n 是前 n 个数的逆元，我们将它乘上 a_n 后，它就会和 a_n 的逆元抵消，从而得到前 $n-1$ 个数的逆元 sv_{n-1}，以此类推，可以计算出所有 sv_i，a_i 的逆元即为 $sv_i \times s_{i-1}$。

```
s[0]=1;
for(int i=1;i<=n;i++) s[i]=s[i-1]*a[i]%p;
inv[n]=qpow(s[n],p-2);
for(int i=n;i>=1;i--) inv[i-1]=inv[i]*a[i]%p;
for(int i=1;i<=n;i++) inv[i]=inv[i]*s[i-1]%p;
```

威尔逊定理

定义：对于质数 p 有 $(p-1)! \equiv -1(\bmod p)$。

证明：https://oi-wiki.org/math/number-theory/wilson/#内容

威尔逊定理

例题：UVA1434

求出下列式子的答案

$$s_n = \sum_{i=1}^{n} \lfloor \frac{(3\times i+6)!+1}{3\times i+7} - \lfloor \frac{(3\times i+6)!}{3\times i+7} \rfloor \rfloor$$

其中，$T \leq 10^3$，$1 \leq n \leq 10^6$

大水紫。

注意到仅当 $3\times i+7$ 为质数时答案才为 1，否则为 0。

$O(n)$ 预处理出来并 $O(1)$ 查询即可。

扩展中国剩余定理

用于求解如下形式的一元线性同余方程组：

$$\begin{cases} x \equiv a_1 \pmod{n_1} \\ x \equiv a_2 \pmod{n_2} \\ \quad \vdots \\ x \equiv a_k \pmod{n_k} \end{cases}$$

例题：P4777，P3868

两个方程的情况

设两个方程分别是 $x \equiv a_1 \pmod{m_1}$、$x \equiv a_2 \pmod{m_2}$;

将它们转化为不定方程:$x = m_1p + a_1 = m_2q + a_2$,其中 p, q 是整数,则有 $m_1p - m_2q = a_2 - a_1$。

由裴蜀定理,当 $a_2 - a_1$ 不能被 $\gcd(m_1, m_2)$ 整除时,无解;

其他情况下,可以通过扩展欧几里得算法解出来一组可行解 (p, q);

则原来的两方程组成的模方程组的解为 $x \equiv b \pmod{M}$,其中 $b = m_1p + a_1$,$M = \text{lcm}(m_1, m_2)$。

多个方程的情况

按之前的方法两两合并即可。

```cpp
typedef long long ll;
typedef __int128 lll;
const int maxn = 1e5 + 5;
void exgcd(ll a, ll b, ll &x, ll &y)
{
    if (b == 0) {x = 1, y = 0; return;}
    exgcd(b, a % b, x, y);
    ll tmp = x;
    x = y; y = tmp - a / b * y;
}
ll n, a[maxn], b[maxn];
void solve()
{
    cin >> n;
    for (int i = 1; i <= n; i++) cin >> a[i] >> b[i];
    ll r = b[1], m = a[1];
    for (int i = 2; i <= n; i++)
    {
        ll d = __gcd(a[i], m), tmp = m, p, q;
        exgcd(m, a[i], p, q);
        m = (lll)a[i] * m / d;
        r = ((lll)tmp * p * (b[i] - r) / d % m + r) % m;
    }
    cout << (r + m) % m << endl;
}
```

习题

- P3951 [NOIP2017 提高组] 小凯的疑惑
- UVA10104 Euclid Problem
- P1516 青蛙的约会
- UVA12775 Gift Dilemma
- P2421 [NOI2002] 荒岛野人
- SP4141 ETF - Euler Totient Function
- UVA11327 Enumerating Rational Numbers
- P2303 [SDOI2012] Longge 的问题
- P2155 [SDOI2008] 沙拉公主的困惑
- P1082 [NOIP2012 提高组] 同余方程
- P4071 [SDOI2016] 排列计数
- P2054 [AHOI2005] 洗牌
- P3868 [TJOI2009] 猜数字
- P2480 [SDOI2010] 古代猪文