数论基础

数论与计数 线性筛 快速线性筛 gcdex 原根 同余方程 逆元 快速幂

欧拉定理 中国剩余定理 mobius反演 扩展bsgs

除法表达式

给出一个这样的除法表达式：X1/X2/X3/.../Xk，其中Xi是正整数。除法表达式应当按照从左到右的顺序求值，例如，表达式1/2/1/2的值为1/4。但可以在表达式中嵌入括号以改变计算顺序，例如，表达式(1/2)/(1/2)的值为1。输入X1,X2,...,Xk，判断是否可以通过添加括号，使表达式的值为整数。K≤10000，Xi≤10^9。

【分析】

1. 除了X_2之外，其他X_i都可以通过添加括号变成分子。
2. 可以用唯一分解定理X_2=，判断每个素数在分子和分母中的次数。
3. 直接约分，使用辗转相除gcd(a,b) = gcd(b, a mod b)。

最小公倍数lcm(a,b) = ab/gcd(a,b)，使用唯一分解定理证明。

无平方因子的数

给出正整数n和m，区间[n,m]内的“无平方因子”的数有多少个？整数p无平方因子，当且仅当不存在k>1，使得p是k^2的倍数。$1 \le n \le m \le 10^{12}$，$m-n \le 10^7$。

【分析】

对于不超过$\sqrt{m}$的所有素数p，筛掉区间[n, m]内p^2的所有倍数。

GCD

- 使用惟一分解定理, 先分解素因数, 然后求最大公约数
- (Euclid算法)利用公式gcd(a, b)=gcd(b, a mod b), O(logb)
- (二进制算法，适合大整数) 若a=b, gcd(a,b)=a, 否则
  - a和b均为偶数, gcd(a,b)=2*gcd(a/2,b/2)
  - a为偶数, b为奇数, gcd(a,b)=gcd(a/2,b)
  - a和b均为奇数, gcd(a,b)=gcd(a-b,b)

扩展欧几里德算法

```
int gcdex(int a, int b, int&x, int& y){
    if(!b){
      x = 1; y = 0; return a;
    }
    int r = gcd(b, a%b, x, y);
    t = x; x = y; y = t – a/b*y;
    return r;
}
```

1. 一定存在整数x,y，使得ax+by=gcd(a,b)
2. 由数学归纳法可证明ax+by=gcd(a,b)
3. 满足ax+by=d的数对(x,y)不是惟一的, 因为当x增加b且y减少a时和不变。
4. 设a,b,c为任意整数，g=gcd(a,b)，方程ax+by=g的一组解是(x_0,y_0)，c是g的倍数则ax+by=c的一组解是$(x_0c/g,y_0c/g)$；否则无整数解。
5. 任意整数解都可以写成$((x_0+k)b/g, (y_0-k)a/g)$，k取任意整数。

例题2.6-8 总是整数(Always an Integer, WF2008, LA4119)

1. 组合数学主要研究计数问题。很多计数问题的答案就是多项式：$(n^2-n)/2$; $(n^4-6n^3+23n^2-18n+24)/24$；$(2n^3+2n^2+n)/6$。当n取任意正整数时，这些多项式的值都是整数。
2. 对于其他多项式，这个性质并不一定成立。
3. 给定一个形如P/D(其中P是n的整系数多项式，D是正整数)的多项式，判断它是否在所有正整数处取到整数值。

【分析】
1. 记最高次数为k，看起来没办法，暴力如何？
2. 其实只需要n=1,2,3…k+1全试一遍即可。
3. F=P/D
4. K=0，计算F (1)即可
5. K=1，F=an+b，就是一个等差数列，只要F(1)和F (2)是整数就行了，F(i)=F(1)+i*(F(2)-F(1))
6. K=2，$F=an^2+bn+c$，Q(n)=F(n+1)-F(n)=2an+a+b，只要Q(1)和Q(2)是整数，Q(n)都是整数，再加上F(1)整数即可
7. K次多项式F(n), 只要保证F(1),F(2),…F(k+1)都是整数即可

例题10-3 选择与除法(Choose and Divide, UVa10375)

已知$C(m,n)=m!/(n!(m-n)!)$，输入整数$p,q,r,s(p\geq q,r\geq s,p,q,r,s\leq 10000)$,计算$C(p,q)/C(r,s)$。输出保证不超过$10^8$，保留5位小数。

【分析】

1. 预处理素数表，分解每一个1~10000
2. 使用唯一分解定理，将乘除法变成素数指数的加减法

例题10-25 约瑟夫的数论 问题（ Joseph' s Problem, NEERC 2005, UVa1363）

输入正整数n和k($1 \le n, k \le 10^9$)，计算$\sum_{i=1}^{n} k \bmod i$。

【分析】

1. 被除数固定，除数逐次加1，余数也应该有规律。
2. 假设k/i的整数部分等于p, 则k mod i=k-i*p。因为k/(i+1)和k/i差别不大，如果k/(i+1)的整数部分也等于p，则k mod (i+1)= k- (i+1)*p = k-i*p-p=k mod i - p。换句话说，如果对于某一个区间i,i+1,i+2,...,j，k除以它们的商的整数部分都相同，则k除以它们的余数会是一个等差数列。
3. 这样，可以在枚举i时把它所在的等差数列之和累加到答案中。这需要计算满足[k/j]=[k/i]=p的最大j。
   1. 当p=0时这样的j不存在，所以等差序列一直延续到序列的最后。
   2. 当p>0时j为满足k/j≥p的最大j，即j≤k/p。
   3. 除了首项之外的项数j-i≤(k-i*p)/p=q/p。

素数筛法

1. Eratosthenes筛法。
   给定外层循环变量i，内层循环的次数n/i 。这样，循环的总次数小于$\sum\frac{n}{i}=O(nlogn)$。这个结论来源于欧拉在1734年得到的结果：$1+\frac{1}{2}+\frac{1}{3}+\ \dots+\frac{1}{n}=\ln(n+1)+\gamma$，其中欧拉常数γ≈0.577218。在很短的时间内得到10^6以内的所有素数。
2. 素数定理$\pi(x)\sim\frac{x}{lnx}$。π(x)表示≤x的素数个数。
3. **快速线性筛-欧拉筛法。**
   1. 任何合数都能表示成一系列素数的积。每个合数必有一个最小素因子，仅被它的最小素因子筛去正好一次。

```
for(int i = 2; i <= n; i++)
  for(int j = i*2; j <= n; j+=i) vis[j] = 1;
```

```
int m = sqrt(n+0.5);
for(int i = 2; i <= m; i++) if(!vis[i])
  for(int j = i*i; j <= n; j+=i) vis[j] = 1;
```

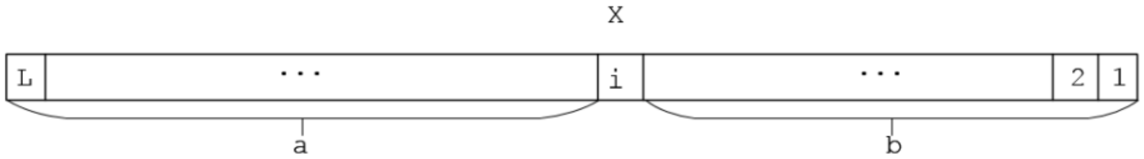
```
for(int i=2;i<n;i++) {
    if(!vis[i])  prime[cnt++]=i;
    for(int j=0;j<cnt && i*prime[j]<n; j++) {
        vis[i*prime[j]]=1;
        if(i%prime[j]==0) break; //关键
    }
}
return cnt;
```

素数判定

- 枚举法: $O(\sqrt{n})$, 指数级别
- 概率算法: Miller-Rabin测试
  - 对于奇数n, 记n=2^rs+1, 其中s为奇数
  - 随机选a(1≤a≤n-1), n如果通过以下的测试，则有很大几率为素数
    - $a^s \equiv 1 (mod\ n)$
    - or 存在0≤j≤r-1使得$a^{2^j s} \equiv -1(mod\ n)$
  - 素数对于所有a通过测试, 合数通过测试的概率不超过1/4
  - 只测试a=2, 3, 5, 7, 则2.5*10^{13}以内唯一个可以通过所有测试的合数为3215031751=(151 * 21291601)
  - 复杂度$O(logN)$

习题10-43整数对(Pair of Integers, ACM/ICPC NEERC 2001, UVa1654)

考虑一个不含前导零的正整数 X，把它去掉一个数字以后得到另外一个数 Y。输入 X+Y 的值 N($1 \le N \le 10^9$)，输出所有可能的等式 X+Y=N。例如 N=34 有两个解：31+3=34；27+7=34。



【分析】

1. 记 n 的 10 进制表示形式的长度为 L。
2. 不妨设从 X 右边数第 i($0 \le i < L$)位删除数字 d($0 \le d < 10$)得到 Y(如上图所示)
3. 记$X = a10^{i+1} + d10^i + b, b < 10^i$, 则 $Y = a10^i + b, X + Y = N$，故有 $11a10^i + d10^i + 2b = N$。
4. 遍历所有的i和x
5. 固定 i 和 x 之后，求不定方程$11a10^i + 2b = N - d10^i$ 所有满足 $a \ge 0$且 $0 \le b < 10^i$ 整数解(a,b)，直接计算并输出 X,Y 即可。

模算术

- 加法
- 减法
- 快速幂
- 大整数取模
  - 把大整数写成“自左向右”的形式
  - 1234=((1*10+2)*10+3)*10+4
  - 依次取模

```
// a^p mod n  0<=a<n
LL pow_mod(LL a, LL p, LL n) {
  if(p == 0) return 1;
  LL ans = pow_mod(a, p/2, n);
  ans = ans * ans % n;
  if(p%2 == 1) ans = ans * a % n;
  return ans;
}
```

线性同余方程ax≡b(mod n)

1. 扩展的Euclid算法
   1. ax-b=ny -> ax-ny=b，求解不定方程即可。
   2. x_0是解的话，所有的x_0+kn都是解。
2. ax-ny=b，$d = \gcd(a, b)$且$d|b$有解，否则无解。
3. 两边同时除以d，得到$a' - n'y = b'$，此时$a' \perp n'$，$a'x \equiv b'(mod\, n)$，$x \equiv a^{\phi(n)-1}b'(mod\, n)$
4. 令$p = a^{\phi(n)-1}b'$，则x = p+kn’都是原方程的解，那到底有多少个mod n意义上的等价类呢? 假如 $p + i \cdot n' \equiv p + j \cdot n'(mod\, n)$ ，则n|$(i - j) \cdot\ n'$。因此$d|(i - j)$，因此刚好有$p + n', p + 2n' \dots . p + (d - 1)n'$个解。

剩余系

1. 模n的完全剩余系就是｛0，1，2，…，n－1｝，常见写法Z/n或者Z_n。
2. 简化剩余系（也称缩系）就是Z_n与n互素的那些元素，记为Z_n*
    1. Z_{12}*={1，5，7，11}。
3. Zn中的每个元素都代表一类元素，其中的运算都是mod n的运算

$ax \equiv 1(\bmod n)$

1. a关于模n的逆（inverse）
2. 类似实数“倒数”。何时a的逆存在?
3. ax-ny=1要有解↔ 1必须是gcd(a,n)的倍数↔a,n互素
4. 此时ax≡1(mod n)只有唯一解，同余方程的解是指一个等价类。
5. Z_{15}中7^{-1} = 13， 3/7 = $3*7^{-1}$ = 3*13 = 9。有两个整数a和b，其中a/b是整数，$a \equiv 3 \ mod\ 15$ 和b $\equiv 7 \ mod\ 15$，则$\frac{a}{b} \equiv 9 \ mod\ 15$，比如a = 528， b = 22。
6. 第二种方法，使用欧拉定理，对于任意a $\in Z_{n*}$, a$^{\phi(n)} \equiv 1(mod\ n)$，因此 $a^{-1} \equiv a^{\phi(n)-1}\ (mod\ n)$

```
LL inv(LL a, LL n) {
  LL d, x, y;
  gcd(a, n, d, x, y);
  return d == 1 ? (x+n)%n : -1;
}
```

欧拉定理

1. 欧拉函数: 1~n中和n互素的元素个数φ(n)
   1. $\varphi(n) = n\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\cdots(1-\frac{1}{p_k})$
2. Euler定理 若gcd(a, n)=1则$a^{\varphi(n)} \equiv 1 \pmod n$
   1. 意义：当b很大时$a^b \equiv a^{b \bmod \varphi(n)} \pmod n$，让指数一直比较小
3. 欧拉函数是积性函数，即当gcd(m,n)=1时f(mn)=f(m)*f(n)

欧拉函数的计算

1. 给定n，需要多少时间计算φ(n)?
   1. $O(\sqrt{n} + \sum p)$, p是n唯一分解中的所有次数之和
   2. 还可以利用素数表来加速
2. 给定n，需要多少时间计算φ(1), φ(2), …, φ(n)的所有值?

```
int euler_phi(int n) {
  int m = (int)sqrt(n+0.5);
  int ans = n;
  for(int i = 2; i <= m; i++) if(n % i == 0) {
    ans = ans / i * (i-1);
    while(n % i == 0) n /= i;
  }
  if(n > 1) ans = ans / n * (n-1);
}
```

```
const int MAXN = 1000000;
int PHI[MAXN + 4];
void phi_table() {
  fill_n(PHI, MAXN + 4, 0);
  PHI[1] = 1;
  _rep(i, 2, MAXN) if (!PHI[i]) {
    for (int j = i; j <= MAXN; j += i) {
      if (!PHI[j]) PHI[j] = j;
      PHI[j] = PHI[j] / i * (i - 1);
    }
  }
}
```

例题2.6-9 最大公约数之和——极限版II(GCDExtreme（II）,UVa11426)

输入正整数n,求gcd(1,2) + gcd(1,3) + gcd(2,3) + ... + gcd(n – 1,n),即所有满足1≤i<j≤n的数对(i,j)所对应的gcd(i,j)之和。比如n = 10时答案为67,n = 100时答案为13015,n = 200000时答案为143295493160。

【分析】

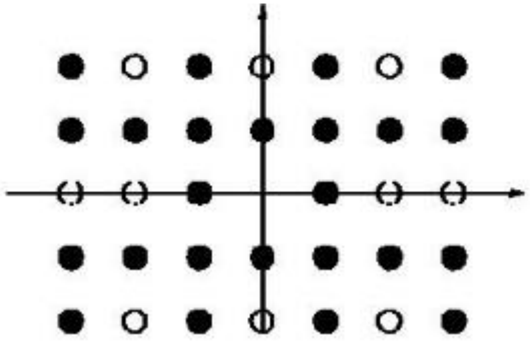
1. 令f(n)=gcd(1,n)+gcd(2,n)+...+gcd(n-1,n)，则所求答案就是$S(\mathrm{n}) = \sum_{i=2}^{n} f(i)$。
2. gcd(x,n)的值都是n的约数，可以按照其值进行分类，g(n,i)=|{x|gcd(x,n)=i, x < n}|，则f(n)=$\sum_{i|n}$i*g(n,i)。
3. gcd(x,n)=i ↔gcd(x/i, n/i)=1，因此满足条件的x/i有φ(n/i)个，也就是说g(n,i)有φ(n/i)个
4. 若依次计算f(n)，需要对n枚举其约数i，速度较慢。反过来对每个i更新其倍数n，类似于素数筛法。

费马小定理

1. 对于任意素数p，有 $a^{p-1} \equiv 1(mod\ p)$
2. 欧拉定理的特殊情况

例题10-27　树林里的树(Trees in a Wood,UVa10214)

在满足|x|≤a,|y|≤b(a≤2000，b≤2000000)的网格中，除了原点之外的整点(即x,y坐标均为整数的点)各种着一棵树。树的半径可以忽略不计，但是可以相互遮挡。求从原点能看到多少棵树。设这个值为K，要求输出K/N，其中N为网格中树的总数。如图所示，只有黑色的树可见。



【分析】

1. 坐标轴上只能看见一棵树，只数第一象限(即x>0，y>0)，答案乘以4后加4
2. 所有x,y都是正整数，能看到(x,y)↔gcd(x,y)=1
3. a范围比较小，b范围比较大，按列统计比较快。
4. 第x列能看到的树的个数等于[0,b)中满足gcd(x,y)=1的y个数。
5. 分区间计算。1≤y≤x: 有ϕ(x)个。x+1≤y≤2x：也有ϕ(x)个，因为gcd(x+i,x)=gcd(x,i)。2x+1≤y≤3x：也有phi(x)个，因为gcd(2x+i,x)=gcd(x,i)。……kx+1≤y≤b：挨个直接统计，需要O(x)时间。
6. 可以预处理ϕ(x)。每次枚举x的所有a种可能，总时间为O(a^2)。

例题10-26 帮帮Tomisu(Help Mr. Tomisu, UVa11440)

给定正整数N和M，统计2和N!之间有多少个整数x满足: x的所有素因子都大于M($2 \le N \le 10^7$, $1 \le M \le N$, $N-M \le 10^5$)。输出答案除以100000007的余数。例如，N=100，M=10时答案为43274465。

【分析】

1. 因为M≤N，所以N!是M!的整数倍。x的所有素因子都大于M↔x⊥M!
2. 根据gcd的性质，对于k>M!，k⊥M! ↔ (k mod M!) ⊥ M!
3. 只需要求出ϕ(M!)，再乘以N!/M!即可
4. 递推预处理出所有的F(n)=ϕ (n!)
5. 由公式$\phi(n)=n\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_1}\right)\cdots(1-\frac{1}{p_k})$：如果n不是素数，那么n!和(n-1)!的素因子集合完全相同，因此F(n)=F(n-1)*n
6. 如果n是素数，那么还会多一项(1-1/n)，即(n-1)/n，约分得F(n)=F(n-1)*(n-1)

中国剩余定理(Chinese Remainder Theorem)

- 考虑方程组$x \equiv a_i (mod\ m_i), m_i \perp m_j$
- 在$0 \le x \le M = m_1 m_2 \cdots m_k$中有唯一解
- 记$w_i = \frac{M}{m_i}$，则$w_i \perp m_i$，存在$p_i, q_i \Rightarrow w_i p_i + m_i q_i = 1$
- 记$e_i = w_i p_i$
  - $j = i \to e_i \equiv 1 (mod\ m_j)$
  - $j \ne i \to e_i \equiv 0 (mod\ m_j)$
- 则原方程等价于 $x \equiv \sum e_i a_i\ (mod\ M)$

```
//n 个方程 x=a[i](mod m[i]) (0<=i<n)
LL china(int n, int* a, int *m) {
  LL M = 1, d, y, x = 0;
  for(int i = 0; i < n; i++) M *= m[i];
  for(int i = 0; i < n; i++) {
    LL w = M / m[i];
    gcd(m[i], w, d, d, y);
    x = (x + y*w*a[i]) % M;
  }
  return (x+M)%M;
}
```

例题2-10 数论难题(Code Feat, UVa11754)

有一个正整数N满足C(1≤C≤9)个条件，每个条件都形如“模X的余数在集合{Y_1,Y_2,...,Y_k}中”，所有的X两两互素，找出最小的S(1≤S≤10)个解。

【输入格式】输入包含若干组数据。每组数据第一行为两个整数C和S(1≤C≤9,1≤S≤10)。以下C行每行描述一个条件，首先是整数X和k(X≥2, 1≤k≤100)，然后是Y_1,Y_2,...,Y_k(0≤ Y_1,Y_2,...,Y_k <X)。所有X的乘积保证在32位带符号整数的范围内。输入结束标志为C=S=0。

【分析】
1. 如果所有k=1，则很容易解决，DFS枚举每个集合中到底那个是余数，套用中国剩余定理。
   1. 若得到的解不超过S个，需要把这些解加上M,2M,3M，直到解足够多（M为所有X的乘积）。
   2. 0不是正整数。 但不要把M,2M,3M... 也忽略 。
2. 但是如果所有k的乘积过大，这个方法就太慢了，考虑选择一个k/X最小的条件，枚举所有符合这个条件的数n，即n=tX+Yi, t=0,1,2....   依次判断n是否符合其它所有条件。
   1. 所有k的乘积很大，这个算法很快就能找到解。

离散对数(BSGS, Shank's Baby-Step-Giant-Step Algorithm)

- 模方程$a^x \equiv b\ (mod\ n)$，先考虑n是素数
- a≠0→存在模意义上的逆a^{-1}
- 因为a^{n-1}≡1(mod n)，x超过n-1就开始循环了，只需检查0,1,2…n-1即可
- 先检查m项, a^0, a^1, … a^{m-1}是否为解，记$e_i=a^i$ mod n，求出a^m的逆a^{-m}
- 考虑a^m, a^{m+1}, … a^{2m-1},如果其中有解，相当于存在i满足$a^{m+i}=e_i a^m \equiv b\ (mod\ n)$，左乘得$e_i \equiv b'$(mod n), $b'=a^{-m}b$。只需要检查是否有$e_i=b'$即可。
- O((m+n/m)logm))，m和n/m接近时m+n/m最小，此时$m=\sqrt{n}$
- (a,n)=1时，上述推导依然成立

```
// 求解a^x≡b (mod n), n为素数，无解返回1
int log_mod(int a, int b, int n) {
  int m, v, e = 1, i;
  m = (int)sqrt(n+0.5);
  v = inv(pow_mod(a, m, n), n);
  map <int,int> x;
  x[1] = 0;
  for(i = 1; i < m; i++){ // 计算e[i]
    e = mul_mod(e, a, n);
    if (!x.count(e)) x[e] = i;
  }
  for(i = 0; i < m; i++){ // 判断 a^im, a^im+1, ..., a^im+m-1
    if(x.count(b)) return i*m + x[b];
    b = mul_mod(b, v, n);
  }
  return -1;
}
```

离散对数(续)

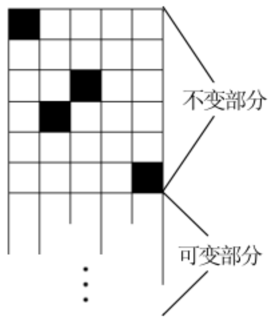
- $(a, n) \neq 1$，设法转化
- $a^x+kn=b$，记$g=(a,n)$，如果$g \nmid b$，则问题无解，否则消去g
- $a'*a^{x-1}+k*n'=b'$，也就是$a^{x'} \equiv b'*(a')^{-1} \pmod{n'}$，其中$a'=a/g$, $n'=n/g$, $b'=b/g$, $x=x'+1$，这样问题规模缩小
- 重复这个过程直到$(a,n)=1$，然后调用BSGS
- 若出现$g \nmid b$，则问题无解。
- 若出现$b=1$， $a^x \equiv 1 \pmod{n}$的解显然为$x=0$，之后递归回去即可

例题2.6-11 网格涂色(Emoogle Grid, UVa11916)

有这样一道题目：要给一个M行N列的网格涂上K种颜色，其中有B个格子不用涂色，其他每个格子涂一种颜色，同一列中的上下两个相邻格子不能涂相同颜色。给出M，N，K和B个格子的位置，求出涂色方案总数除以10^8+7的余数R。本题的任务和这个相反：已知N，K，R和B个格子的位置，求最小可能的M。

【分析】

1. 按列涂色，每列从上往下，一个格子位于第一行或者上面格子不能涂色，它有K种，其它有K-1种。M未知，但是M≥(不能涂色格子编号的最大值)，可将网格分成不变和可变两部分。
2. 假定不变部分以及可变部分第一行(加起来有L行)共有C种方案，每加一行总方案数*=$(K-1)^N$,记$P=(K-1)^N$
3. 这样得到一个模方程$C\times P^x \equiv R$，$P^x \equiv R\times C^{-1}$。
4. 注意要判断完全不加新行是否能满足要求。



原根

- 对于素数p，如果存在一个正整数1<a<p，使得$a^1,a^2,a^3,\ldots,a^{p-1}$模p的值取遍1,2,…,p-1的所有整数，称a是p的一个原根(primitive root)
- 不难发现$a^1,a^2,a^3,\ldots,a^{p-1}$和1,2,…p-1的值必须一一对应
- 根据欧拉定理$a^{p-1} \equiv 1 \pmod p$，$a^{p+i} \equiv a^{i+1}$, $a^i \equiv a^j \pmod p \rightarrow i \equiv j \pmod{p-1}$
  - 3是7的原根，因为3→2→6→4→5→1[22]，然后开始循环：1→3→2→6…
  - 2不是7的原根，因为2→4→1→2→4->1…，过早的循环了。
  - 注意到 $a^{p-1} \equiv 1 \pmod p$，这个生成序列一定会包含1，且在此之前不会有循环——要是在出现1之前就循环了，就永远不会出现1了，与欧拉定理矛盾。
- 原根可以证明有$\phi(p-1)$个，可以逐一判断，如何判断一个数m是否原根:
  - 枚举循环节长度b，也就是p-1的所有真因子,如果有$m^b \equiv 1 \pmod p$,表示m非原根。
- 并不是只有素数才有原根，而且高次方程也不是只有当模有原根的时候才能解。

例题6-20 信息解密(Decrypt Messages, Shanghai 2009, LA 4746)

假设从2000年1月1日00:00:00到现在经过了x秒，计算x^q mod p，设答案为a。这里素数p严格大于x。已知p，q，a，求现在时刻的所有可能值。

提示：如果一个年份是4的倍数但不是100的倍数，或者这个年份是400的倍数，这个年份是闰年。闰年的二月有29天，其他年（平年）的二月只有28天。在本题中，如果年份除以10的余数为5或者8，则这一年的最后还会有一个"闰秒"。比如，2005年12月31日23:59:59的下一秒是2005年12月31日23:59:60，再下一秒才是2006年1月1日00:00:00。

【分析】

1. 除了日历部分，实际上是要解高次模方程$x^q \equiv a \pmod p$。
2. 找到p的一个原根m，设$x \equiv m^y$,$a \equiv m^z$，则方程变为$m^{qy} \equiv m^z \pmod p$
3. $qy \equiv z \pmod{p-1}$，而$m^z \equiv a$，利用bsgs可以求出z，然后再求出y
4. 最后使用pow_mod得到x即可

积性函数与Möbius函数

- 考虑ϕ(m)，$m=m_1m_2 \; m_1 \perp m_{2\%}$
- 则$n \perp m \leftrightarrow n \perp m_1$且$n \perp m_2$，根据中国剩余定理n与二元组(n mod m1, n mod m2)一一对应，ϕ(n)是积性函数。
- $g(n) = \sum d|n \; f(d), n \geq 1$，则f是积性函数↔g是积性函数
- 考虑φ的容斥原理推导，$\phi(n) = n - \sum_{S \subseteq \{p_1,p_2,\cdots p_k\}} (-1)^{|S|} \cdot \frac{n}{\prod_{p_i \in S} p_i}$
- 遍历n的所有因子d(不一定是素数)，设因子d的系数为μ(d)：
  - $\mu(d) = \begin{cases} 1, \; n = 1 \\ (-1)^k, \; n = p_1p_2\cdots p_k, \; \mathrm{p_i} \neq pj \\ 0, \; otherwise \end{cases}$
- 则$\phi(n) = \sum_{d|u} \mu(d) \frac{n}{d}$，上述μ就是MÖbius函数，phi函数的上述写法就是一个Möbius反演

Möbius反演

- n为正整数
  - $g(n) = \sum_{\mathrm{d}|n} f(d) \rightarrow f(n) = \sum_{\mathrm{d}|n} \mu(d) g\left(\frac{n}{d}\right)$
  - $g(n) = \sum_{n|d} f(d) \rightarrow f(n) = \sum_{n|d} \mu\left(\frac{d}{n}\right) g(d)$

数一数a*b(Count a*b, ACM/ICPC Changchun 2015, LA7184)

- 令f(m)表示有多少个二元组(a,b)满足：a和b都是小于m的非负整数，且ab不是m的倍数。输入n($1 \le n \le 10^9$)，求$g(n)=\sum_{m|n} f(m)$。
- 比如g(6)=f(1)+f(2)+f(3)+f(6)=0+1+4+21=26，g(514)=328194。
- 一共有T组数据，*1≤T≤20000,* 输出答案除以2^{64}的余数。

LA3571 Visible Lattice Points(Greater New York 2006)

- 对于二维坐标系中的点{(x,y)(0 <= x <= N, 0 <= y <= N)}，问这些点中有哪些是原点可以看到的。(原点除外)

【分析】

1. 可以转化为求gcd(x, y) == 1 的有序对(x, y)的数目。
2. 定义$f(d)=|\{x,y|gcd(x,y)=d\}, x,y\le N|$, $g(d)=|\{x,y| d|gcd(x,y)\}, x,y\le N|$，显然$g(d)=(N/d)^2$
3. 显然$g(n)=\sum_{n|d}f(d)$
4. 反演得: $f(n)=\sum_{n|d}\mu(d/n)g(d)= \sum_{n|d}\mu(d/n)(N/d)^2$