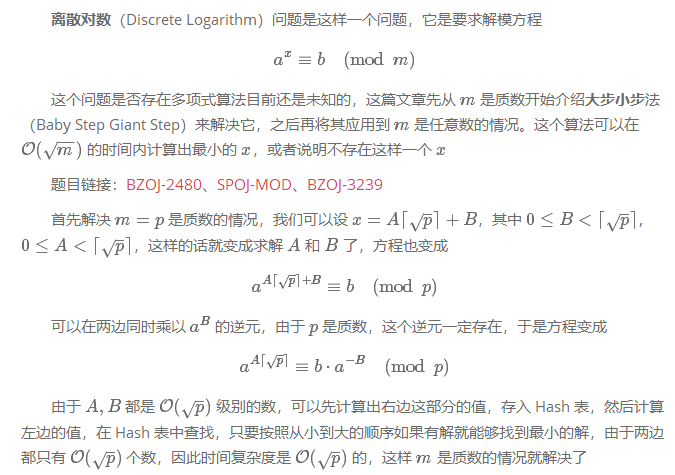
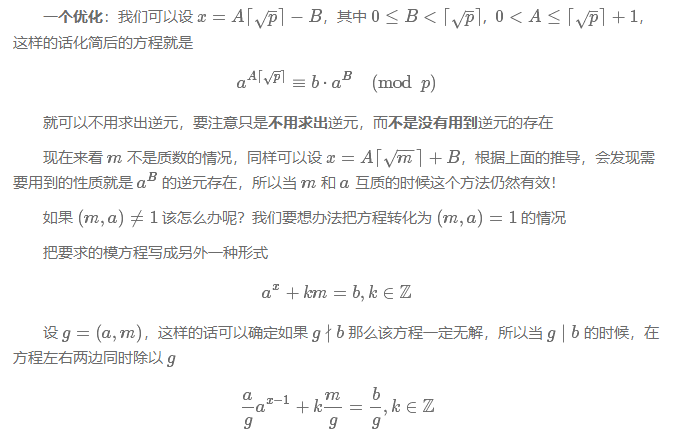
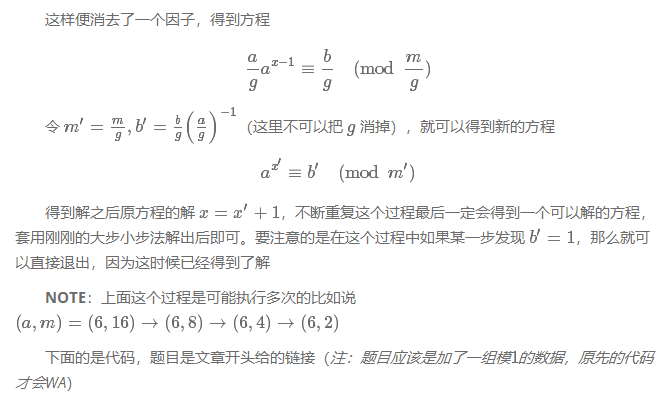
# **扩展大步小步法解决离散对数问题**







/\* BZOJ-2480: Spoj3105 Mod

\* 扩展大步小步 \*/

#include <cstdio>

#include <cmath>

#include <map>

int mod\_pow(long long x, long long p, long long mod\_v)

{

long long v = 1;

while(p)

{

if(p & 1) v = x \* v % mod\_v;

x = x \* x % mod\_v;

p >>= 1;

}

return v;

}

int gcd(int a, int b)

{

return b ? gcd(b, a % b) : a;

}

int baby\_step\_giant\_step(int a, int b, int p)

{

a %= p, b %= p;

if(b == 1) return 0;

int cnt = 0;

long long t = 1;

for(int g = gcd(a, p); g != 1; g = gcd(a, p))

{

if(b % g) return -1;

p /= g, b /= g, t = t \* a / g % p;

++cnt;

if(b == t) return cnt;

}

std::map<int, int> hash;

int m = int(sqrt(1.0 \* p) + 1);

long long base = b;

for(int i = 0; i != m; ++i)

{

hash[base] = i;

base = base \* a % p;

}

base = mod\_pow(a, m, p);

long long now = t;

for(int i = 1; i <= m + 1; ++i)

{

now = now \* base % p;

if(hash.count(now))

return i \* m - hash[now] + cnt;

}

return -1;

}

int main()

{

int a, b, p;

while(std::scanf("%d %d %d", &a, &p, &b), p)

{

int ans = baby\_step\_giant\_step(a, b, p);

if(ans == -1) std::puts("No Solution");

else std::printf("%d\n", ans);

}

return 0;

}