```
lehmer_pi
#include<bits/stdc++.h>
using namespace std;
typedef long long LL;
using namespace std;
const int N = 5e6 + 2;
bool np[N];
int prime[N], pi[N];
const LL MAX = 22900000000LL;
int getprime()
{
    int cnt = 0;
    np[0] = np[1] = true;
    pi[0] = pi[1] = 0;
    for(int i = 2; i < N; i++)
    {
        if(!np[i]) prime[++cnt] = i;
        pi[i] = cnt;
        for(int j = 1; j <= cnt && i * prime[j] < N; j++)
        {
            np[i * prime[j]] = true;
            if(i % prime[j] == 0)   break;
        }
    }
    return cnt;
}
const int M = 7;
const int PM = 2 * 3 * 5 * 7 * 11 * 13 * 17;
int phi[PM + 1][M + 1], sz[M + 1];
void Init()
{
    getprime();
    sz[0] = 1;
    for(int i = 0; i <= PM; i++)
        phi[i][0] = i;
    for(int i = 1; i <= M; i++)
    {
        sz[i] = prime[i] * sz[i - 1];
        for(int j = 1; j <= PM; j++)
            phi[j][i] = phi[j][i - 1] - phi[j / prime[i]][i - 1];
    }
}
int sqrt2(LL x)
{
    LL r = (LL)sqrt(x - 0.1);
    while(r * r <= x)   ++r;
    return int(r - 1);
}
int sqrt3(LL x)
{
    LL r = (LL)cbrt(x - 0.1);
    while(r * r * r <= x)   ++r;
    return int(r - 1);
}
```

```
LL get_phi(LL x, int s)
{
    if(s == 0)  return x;
    if(s <= M)  return phi[x % sz[s]][s] + (x / sz[s]) * phi[sz[s]][s];
    if(x <= prime[s]*prime[s])   return pi[x] - s + 1;
    if(x <= prime[s]*prime[s]*prime[s] && x < N)
    {
        int s2x = pi[sqrt2(x)];
        LL ans = pi[x] - (s2x + s - 2) * (s2x - s + 1) / 2;
        for(int i = s + 1; i <= s2x; i++) ans += pi[x / prime[i]];
        return ans;
    }
    return get_phi(x, s - 1) - get_phi(x / prime[s], s - 1);
}
LL getpi(LL x)
{
    if(x < N)   return pi[x];
    LL ans = get_phi(x, pi[sqrt3(x)]) + pi[sqrt3(x)] - 1;
    for(int i = pi[sqrt3(x)] + 1, ed = pi[sqrt2(x)]; i <= ed; i++)
        ans -= getpi(x / prime[i]) - i + 1;
    return ans;
}
LL lehmer_pi(LL x)
{
    if(x < N)   return pi[x];
    int a = (int)lehmer_pi(sqrt2(sqrt2(x)));
    int b = (int)lehmer_pi(sqrt2(x));
    int c = (int)lehmer_pi(sqrt3(x));
    LL sum = get_phi(x, a) + (LL)(b + a - 2) * (b - a + 1) / 2;
    for (int i = a + 1; i <= b; i++)
    {
        LL w = x / prime[i];
        sum -= lehmer_pi(w);
        if (i > c) continue;
        LL lim = lehmer_pi(sqrt2(w));
        for (int j = i; j <= lim; j++)
            sum -= lehmer_pi(w / prime[j]) - (j - 1);
    }
    return sum;
}
```

等差数列异或和

```
LL Get(LL dis, LL l, LL P, LL number)
{
    LL ret = 0;
    ret += (l / P) * number;
    l %= P;
    ret += (dis / P) * number * (number - 1) / 2;
    dis %= P;
    if (dis * number + l < P)
        return ret;
    else
        return ret + Get(P, (dis * number + l) % P, dis, (dis * number + l)
/ P);
}

LL GetYiHuo(LL l, LL r, LL dis)  //以x开始， y为结束， dis为等差 连续异或
{
    //number为计算的个数
    LL number = (r - l) / dis + 1, ans = 0, Sum, P = 1;
    for (LL i = 1; i <= 10; i++)
    {
        Sum = Get(dis, l, P, number);
        if (Sum & 1)
            ans += P;
        P <<= 1;
    }
    return ans;
}
```

```
g++  main.cpp  -o  main  -std=c++11  &>warning.txt  &&  ./main<in.txt
这条语句是把编译信息写入 warning.cpp
~warning.txt
```