# Min_25 new model

```cpp
#include<bits/stdc++.h>
using namespace std;
namespace MIN25{
    const int N = 5 + 2e5;//2 * sqrt_n
    const long long mod = 7 + 1e9;
    bool np[N];
    long long n;
    int sqrt_n, m, tot, pri[N], id0[N], id1[N];
    //int f_t, g_t; 根据f(p)表达式里的p的幂次来创建f_t和g_t,比如f(p)=p-1,f_1[p]=p, f_2[p]=1, f(p)=f_1[p]-f_2[p],g_t同理
    //Fp_t=\sum_{p}f_t
    long long w[N];
    long long f(long long p, long long t) { // f(p ^ t)
        //...
    }
    void init(int n) {
        tot = 0;
        np[1] = 1;
        for (int i = 2; i <= n; i++) {
            if (!np[i]) {
                np[i] = 1; pri[++tot] = i;
                //calc Fp_t[tot] = Fp_t[tot-1]+f_t[i]
            }
            for (int j = 1, val = i * pri[j]; j <= tot && val <= n; j++) {
                np[val] = 1;
                if (i % pri[j] == 0) { break; }
            }
        }
    }

    int get_id(long long x) {
        if (x <= sqrt_n) return id0[x];
        else return id1[n / x];
    }
    void sieve_g(long long n) {
        m = 0;
        for (long long i = 1, j; i <= n; i = j + 1) {
            long long k = n / i; j = n / k;
            w[++m] = k;
            if (k <= sqrt_n) id0[k] = m;
            else id1[n / k] = m;
            k %= mod;
            //calc g_t(m) = \sum_{j=2}^w[m]f_t(j)
        }
        for (int i = 1; i <= tot; i++) {
            for (int j = 1; j <= m && 1ll * pri[i] * pri[i] <= w[j]; j++) {
                int id = get_id(w[j] / pri[i]);
                //translation g_t[j]=(g_t[j]-(f_t(p[i])*(g_t[op]-Fp_t[i-1]+MOD)%MOD)%MOD+MOD)%MOD
                //即:g_t[j]-=f_t(p[i])*(g_t[op]-sum_t[i-1])
            }
        }
```

```
    }
    long long S(long long x, long long y) {
        if (x <= 1 || pri[y] > x) return 0;
        long long id = get_id(x), res = 0;
        //质数部分:res = g_t[id] - Fp_t[y - 1]
        for (int i = y; i <= tot && 1ll * pri[i] * pri[i] <= x; i++) {
            long long t0 = pri[i], t1 = pri[i] * pri[i];
            for (long long e = 1; t1 <= x; t0 = t1, t1 *= pri[i], e++) {
                long long fp0 = f(pri[i], e), fp1 = f(pri[i], e + 1); // 此处
可优化
                res += (fp0 * S(x / t0, i + 1) % mod + fp1) % mod;
                res %= mod;
            }

        }
        return res;
    }
    long long solve(long long n) {
        sqrt_n = 2ll * sqrt(n) + 1; // 2 * sqrt_n ~ n^{2/3}
        init(sqrt_n); seive_g(n);
        //f1 = ... f(1)的值
        return (S(n, 1) + f1) % mod;
    }
}
signed main() {

}
```