// 三个线性筛

const int MAXN=1e7;

int prime[MAXN+48],tot;bool isprime[MAXN+48];

int phi[MAXN+48],mu[MAXN+48];

// 线性筛素数

void sieve\_prime()

{

memset(isprime,true,sizeof(isprime));

isprime[1]=false;

int i,j;

for (i=2;i<=n;i++)

{

if (isprime[i]) prime[++tot]=i;

for (j=1;j<=tot && 1ll\*i\*prime[j]<=n;j++)

{

isprime[prime[j]\*i]=false;

if(i%prime[j]==0) break;

}

}

}

// 线性筛欧拉函数

void sieve\_phi()

{

memset(isprime,true,sizeof(isprime);

int i,j;

for (i=2;i<=n;i++)

{

if (isprime[i]) prime[++tot]=i,phi[i]=i-1;

for (j=1;j<=tot && 1ll\*prime[j]\*i<=n;j++)

{

isprime[i\*prime[j]]=false;

if (i%prime[j]==0) phi[i\*prime[j]]=(phi[i]\*prime[j])%MOD; else phi[i\*prime[j]]=(phi[i]\*(prime[j]-1))%MOD;

if (i%prime[j]==0) break;

}

}

}

// 线性筛莫比乌斯函数

void sieve\_mu()

{

int i;

memset(isprime,true,sizeof(isprime));

for (i=2;i<=n;i++)

{

if (isprime[i]) prime[++tot]=i,mu[i]=-1;

for (j=1;j<=tot && 1ll\*prime[j]\*i<=n;j++)

{

isprime[i\*prime[j]]=false;

if (i%prime[j]==0) mu[i\*prime[j]]=0; else mu[i\*prime[j]]=-mu[i];

if (i%prime[j]==0) break;

}

}

}