问题描述（洛谷4213）

给定N($N<2^{31}$)求1到N的欧拉函数和莫比乌斯函数之和

解决方法

定义

对函数f(n),g(n)定义$(f*g)(n)=\sum_{d\mid n}f(d)g(\frac{n}{d})$为f与g的狄利克雷卷积

引理1

$\sum_{i=1}^n(f*g)(i)=\sum_{i=1}^n\sum_{d\mid i}f(d)g(\frac{i}{d})=\sum_{d=1}^n f(d)\sum_{i=1}^{\lfloor \frac{n}{d}\rfloor}g(i)=\sum_{d=1}^nf(d)S(\lfloor \frac{n}{d}\rfloor)$其中$S(n)=\sum_{i=1}^n g(i)$

引理2

设$f(n)=1,g(n)=\phi (n)$则有$(f*g)(n)=n$

引理3

设$f(n)=1,g(n)=\mu (n)$则有$(f*g)(n)=[n=1]$

具体解决

由引理1得$f(1)S(n)=\sum_{i=1}^n(f*g)(i)-\sum_{d=2}^nf(d)S(\lfloor \frac{n}{d}\rfloor)$

分别将$f(n)=1,g(n)=\phi (n)$和$f(n)=1,g(n)=\mu (n)$带入，得

$S(n)=\frac{n^2+n}{2}-\sum_{d=2}^nS(\lfloor \frac{n}{d}\rfloor)$其中$S(n)=\sum_{i=1}^n \phi (i)$

$T(n)=1-\sum_{d=2}^nT(\lfloor \frac{n}{d}\rfloor)$其中$T(n)=\sum_{i=1}^n \mu (i)$

先线性求出10^7以内的$S(n),T(n)$对于大于10^7的$S(n),T(n)$可通过数论分块递归计算出。

可以证明，杜教筛的时间复杂度为$O(n^{\frac{3}{4}})$

```cpp #include&lt;iostream&gt; #include&lt;cstdio&gt; #include&lt;map&gt; #define N 7000001 #define LL long long using namespace std;

LL T,n,phi[N],mo[N],p[N»3]; map&lt;LL,LL&gt;sphi,smo; void initial() {

```
mo[1]=1; phi[1]=1;
```
```

```
for(int i=2;i<N;i++)
{
    if(!phi[i])
    {
        mo[i]=-1;
        phi[i]=i-1;
        p[++p[0]]=i;
    }
    for(int j=1;;j++)
    {
        LL t=i*p[j];
        if(t>=N) break;
        if(i%p[j]==0)
        {
            mo[t]=0;
            phi[t]=phi[i]*p[j];
            break;
        }
        mo[t]=-mo[i];
        phi[t]=phi[i]*(p[j]-1);
    }
}
for(int i=2;i<N;i++)
{
    mo[i]+=mo[i-1];
    phi[i]+=phi[i-1];
}
```

} LL getphi(LL x) {

```
if(x<N) return phi[x];
if(sphi[x]) return sphi[x];
LL res=x*(x+1)>>1;
for(LL l=2,r;l<=x;l=r+1)
{
    r=x/(x/l);
    res-=(r-l+1)*getphi(x/l);
}
sphi[x]=res;
return res;
```

} LL getmo(LL x) {

```
if(x<N) return mo[x];
if(smo[x]) return smo[x];
LL res=1;
for(LL l=2,r;l<=x;l=r+1)
{
    r=x/(x/l);
```

```
    res-=(r-l+1)*getmo(x/l);
}
smo[x]=res;
return res;
```

} int main() {

```
initial();
scanf("%lld",&T);
while(T--)
{
    scanf("%lld",&n);
    printf("%lld ",getphi(n));
    printf("%lld\n",getmo(n));
}
return 0;
```

} ```