

1

数论 2

前置公式

$$\begin{aligned} \left(\sum_{d \mid n} f(d) \right) \left(\sum_{d \mid m} g(d) \right) &= \sum_{d_1 \mid n} \sum_{d_2 \mid m} f(d_1)g(d_2) \end{aligned}$$

$$\sum_{m=1}^n \sum_{d \mid (n,m)} f(d) = \sum_{d \mid n} \frac{f(d)n}{d}$$

$$\sum_{n=1}^{\infty} \sum_{x \mid n} \left(f(x) \sum_{y \mid \frac{n}{x}} g(y) \right) = \sum_{y \mid n} \left(g(y) \sum_{x \mid \frac{n}{y}} f(x) \right)$$

可积函数

一般可积函数

定义

数论函数 θ 被定义为可积函数，若

$$(a) \exists n \text{ s.t. } \theta(n) \neq 0$$

$$(b) \forall n \forall m \text{ s.t. } (n,m)=1 \Rightarrow \theta(nm) = \theta(n)\theta(m)$$

性质

假定 θ 为可积函数，则

$$\begin{aligned} &\theta(1)=1 \\ &\theta(n) = \theta(\prod_{i=1}^k p_i^{\alpha_i}) = \prod_{i=1}^k \theta(p_i^{\alpha_i}) = \prod_{i=1}^k \theta(p_i)^{\alpha_i} \end{aligned}$$

定理

若 θ 为可积函数，则

$$\begin{aligned} \psi(n) &= \sum_{d \mid n} \theta(d) \\ \psi(n) &= \prod_{i=1}^k (1 + \theta(p_i)) \end{aligned}$$

证明

设 $(n, m) = 1$ 则对 $d \mid nm$ 一定存在唯一 $d_1 \mid n, d_2 \mid m, d = d_1 d_2$

同时，对每对 $d_1 \mid n, d_2 \mid m$ 一定有 $d = d_1 d_2 \mid nm$

所以根据 \$(1)\$ 式

$$\begin{aligned} \psi(nm) &= \sum_{d \mid nm} \theta(d) = \sum_{d_1 \mid n} \sum_{d_2 \mid m} \theta(d_1 d_2) \\ \theta(d_1) \theta(d_2) &= \left(\sum_{k \mid n} \theta(k) \right) \left(\sum_{l \mid m} \theta(l) \right) = \psi(n) \psi(m) \end{aligned}$$

$$\begin{aligned} \psi(n) &= \prod_{i=1}^k \psi(p_k) = \prod_{i=1}^k \left(1 + \theta(p_k) + \theta(p_k^2) + \dots + \theta(p_k^{\alpha_k}) \right) \\ &= \prod_{i=1}^k \left(1 - \frac{1}{p_k} \right) \tag{4} \end{aligned}$$

莫比乌斯函数

定义

$$\mu(n) = \begin{cases} 1 & n=1 \\ (-1)^k & n=p_1 p_2 \dots p_k \\ 0 & \text{otherwise} \end{cases}$$

性质

若 θ 为可积函数，根据 \$(4)\$ 式

$$\begin{aligned} \sum_{d \mid n} \mu(d) \theta(d) &= \prod_{i=1}^k \left(1 - \frac{1}{p_k} \right) \theta(p_k) + \prod_{i=1}^{k-1} \left(1 - \frac{1}{p_k} \right) \theta(p_k^2) + \dots + \theta(p_k^{\alpha_k}) \\ &= \prod_{i=1}^k \left(1 - \frac{1}{p_k} \right) \theta(p_k) = \mu(p_k) \theta(p_k) = \mu(p_k) \end{aligned}$$

特别地

$$\theta(n) \equiv 1 \iff \sum_{d \mid n} \mu(d) = 1 \iff n = 1$$

$$\theta(n) = \frac{1}{n} \sum_{d \mid n} \frac{\mu(d)}{d} = \frac{(1-p_1)(1-p_2)\dots(1-p_k)}{n}$$

欧拉函数

定义

$$\varphi(n) = \sum_{m=1}^n [(n, m) == 1]$$

性质

根据 \$(2)\$ 式、\$(6)\$ 式和 \$(7)\$ 式

$$\begin{aligned} \varphi(n) &= \sum_{m=1}^n [n,m]=1 = \sum_{m=1}^n \sum_{d|m} \mu(d) \\ d &= n \sum_{d|m} \frac{\mu(d)}{d} = n(1-p_1)(1-p_2)\dots(1-p_k) \end{aligned}$$

其他常见可积函数

$$\tau(n) = \sum_{d|m} 1 = \prod_{i=1}^k (\alpha_k + 1)$$

$$\sigma(n) = \sum_{d|m} d = \prod_{i=1}^k (1 + p_k + p_k^2 + \dots + p_k^{\alpha_k})$$

完全积性函数

$f(x)$ 被定义为完全积性函数若 $\forall n \forall m (f(nm) = f(n)f(m))$

$$e(n) = [n == 1]$$

$$I(n) = 1$$

$$id(n) = n$$

线性筛法

先给出线性筛素数的代码

```
const int MAXP=1e6;
bool vis[MAXP];
int prime[MAXP], cnt;
void Prime(){
    vis[1]=true;
    for(i,2,MAXP){
        if(!vis[i]) prime[cnt++]=i;
        for(int j=0;j<cnt&&i*prime[j]<MAXP;j++){
            vis[i*prime[j]]=true;
            if(i%prime[j]==0) break;
        }
    }
}
```

基于线性筛素数和可积函数性质，如果能在 $O(1)$ 时间求出 $f(p^k)$ 就可以线性筛 f 函数，例如

```
const int MAXP=1e6;
bool vis[MAXP];
```

```
int prime[MAXP],mu[MAXP],cnt;
void Mu(){
    vis[1]=true,mu[1]=1;
    _for(i,2,MAXP){
        if(!vis[i])mu[i]=-1,prime[cnt++]=i;
        for(int j=0;j<cnt&&i*prime[j]<MAXP;j++){
            vis[i*prime[j]]=true;
            if(i%prime[j])
                mu[i*prime[j]]=-mu[i];
            else{
                mu[i*prime[j]]=0;
                break;
            }
        }
    }
}
```

```
const int MAXP=1e6;
bool vis[MAXP];
int prime[MAXP],phi[MAXP],cnt;
void Phi(){
    vis[1]=true,phi[1]=1;
    _for(i,2,MAXP){
        if(!vis[i])phi[i]=i-1,prime[cnt++]=i;
        for(int j=0;j<cnt&&prime[j]*i<MAXP;j++){
            vis[i*prime[j]]=true;
            if(i%prime[j])
                phi[i*prime[j]]=phi[i]*(prime[j]-1);
            else{
                phi[i*prime[j]]=phi[i]*prime[j];
                break;
            }
        }
    }
}
```

```
const int MAXP=1e6;
bool vis[MAXP];
int prime[MAXP],tau[MAXP],mpow[MAXP],cnt;
void Tau(){
    vis[1]=true,tau[1]=1;
    _for(i,2,MAXP){
        if(!vis[i])tau[i]=2,mpow[i]=1,prime[cnt++]=i;
        for(int j=0;j<cnt&&prime[j]*i<MAXP;j++){
            vis[i*prime[j]]=true;
            if(i%prime[j])
                tau[i*prime[j]]=tau[i]<<1,mpow[i*prime[j]]=1;
            else{
```

```

tau[i*prime[j]]=tau[i]/(mpow[i]+1)*(mpow[i]+2),mpow[i*prime[j]]=mpow[i]+1;
        break;
    }
}
}

```

简单地说，

$\begin{aligned} f(ip) = \begin{cases} f(i)f(p) & p \text{ 不是 } i \text{ 的最小素因子} \\ f(p^{k+1})/f(p^k) & p \text{ 是 } i \text{ 的最小素因子} \end{cases} \end{aligned}$

如有必要，额外存储每个数的最小素因子的幂次即可。

狄利克雷卷积

定义

两个数论函数 f 和 g 的迪利克雷卷积运算为 $(f \ast g)(n) = \sum_{d \mid n} f(d)g(\frac{n}{d})$

性质

- 1、交换律 $f \ast g = g \ast f$
- 2、结合律 $(f \ast g) \ast h = f \ast (g \ast h)$
- 3、分配律 $(f+g) \ast h = f \ast h + g \ast h$
- 4、单位元 $e \ast f = f$
- 5、逆元：对每个 $f(1) \neq 0$ 的数论函数 f 一定存在某个数论函数 g 使得 $f \ast g = e$
- 6、两个可积函数的迪利克雷卷积仍为可积函数，可积函数的逆元也为可积函数。

性质 5 证明

事实上，构造 $g(n) = \frac{1}{f(1)} \left(e(n) - \sum_{d \mid n, d \neq 1} f(d)g(\frac{n}{d}) \right)$

知 $g(n)$ 可由 $g(1 \sim n-1)$ 递推得到，且 $\sum_{d \mid n} f(d)g(\frac{n}{d}) = f(1)g(n) + \sum_{d \mid n, d \neq 1} f(d)g(\frac{n}{d}) = e(n)$

莫比乌斯反演定理

设 $F(x) = f(x)$ 为数论函数 $F(n) = \sum_{d \mid n} f(d) \iff f(n) = \sum_{d \mid n} \mu(n/d)F(d)$

(d) $F(\frac{n}{d})\sum_{d|n} f(d)$

该定理的另一种形式为 $\sum_{n \leq x} F(n) = \sum_{d \leq x} \mu(d) F(\frac{x}{d})$ iff $\sum_{n \leq x} f(n) = \sum_{d \leq x} \mu(d) f(\frac{x}{d})$

事实上，该定理等价于若 $F = f \ast \mu$ 则 $f = F \ast \mu$ 。简单地说，莫比乌斯反演定理即证明 $e = \mu \ast \mu$

证明

根据 \$(3)\$ 式和 \$(6)\$ 式

$$\begin{aligned} \sum_{d|n} \mu(d) F\left(\frac{n}{d}\right) &= \sum_{d|n} \left(\sum_{d_1|d} \mu(d_1) f\left(\frac{d}{d_1}\right) \right) \\ &= \sum_{d_1|n} f(d_1) \sum_{d_2|n/d_1} \mu(d_2) \end{aligned}$$

一些常见的狄利克雷卷积

$$1. \mu \ast \mu = e$$

$$2. \mu \ast id = \varphi$$

$$3. \mu \ast id = \sigma$$

$$4. \mu \ast I = \tau$$

$$5. \varphi \ast id = id$$

$$6. \tau \ast \varphi = \sigma$$

证明 1

莫比乌斯反演定理已证明。

证明 2

欧拉函数性质推导过程已证明。

证明 3

根据定义。

证明 4

根据定义。

证明 5

根据前面结论 $\ast \varphi = \ast (\mu \ast id) = (\ast \mu) \ast id = e \ast id = id$

证明 6

根据前面结论 $\ast \tau \varphi = (\ast I) \ast (\mu \ast id) = (\ast \mu) \ast (\ast id) = e \ast \sigma = \sigma$

算法练习

习题1

[洛谷p3455](#)

题意

给定 a, b, d 求满足 $x \leq a, y \leq b, (x, y) = d$ 的二元组。

题解 1

在约束条件 $i \leq a, j \leq b$ 下，设 $f(n)$ 为满足 $(i, j) = n$ 的二元组个数 $F(n)$ 为满足 $n \mid (i, j)$ 的二元组个数。

$$\begin{aligned} f(n) &= \sum_{i=1}^a \sum_{j=1}^b [(i, j) == n] \end{aligned}$$

$$\begin{aligned} F(n) &= \sum_{i=1}^a \sum_{j=1}^b [n \mid (i, j)] = \lfloor \frac{an}{n} \rfloor - \lfloor \frac{bn}{n} \rfloor \end{aligned}$$

根据莫比乌斯反演定理，有

$$\begin{aligned} Ans &= f(d) = \sum_{d \mid k} \sum_{k \leq \min(a, b)} \mu(\lfloor \frac{kd}{n} \rfloor) F(k) \\ &= \sum_{d \mid k} \sum_{k \leq \min(a, b)} \mu(\lfloor \frac{kd}{n} \rfloor) \left(\lfloor \frac{ak}{n} \rfloor - \lfloor \frac{bk}{n} \rfloor \right) \end{aligned}$$

设 $k = t \ast d$ 改为枚举 t 有

$$\begin{aligned} Ans &= \sum_{d \mid k} \sum_{k \leq \min(a, b)} \mu(\lfloor \frac{kd}{n} \rfloor) \left(\lfloor \frac{ak}{n} \rfloor - \lfloor \frac{bk}{n} \rfloor \right) \\ &= \sum_{t=1}^{\min(a, b)} \sum_{d \mid t} \mu(t) \left(\lfloor \frac{at}{n} \rfloor - \lfloor \frac{bt}{n} \rfloor \right) \end{aligned}$$

剩下部分数论分块即可，时间复杂度 $O(\sqrt{n})$

```
#include <cstdio>
```

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <cstring>
#include <cctype>
#include <queue>
#include <cmath>
#define _for(i,a,b) for(int i=(a);i<(b);++i)
#define _rep(i,a,b) for(int i=(a);i<=(b);++i)
#define mem(a,b) memset((a),(b),sizeof(a))
using namespace std;
typedef long long LL;
inline int read_int(){
    int t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline LL read_LL(){
    LL t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline void write(LL x){
    register char c[21],len=0;
    if(!x) return putchar('0'),void();
    if(x<0)x=-x,putchar('-');
    while(x)c[++len]=x%10,x/=10;
    while(len)putchar(c[len--]+48);
}
inline void space(LL x){write(x),putchar(' ');}
inline void enter(LL x){write(x),putchar('\n');}
const int MAXP=5e4+5;
bool vis[MAXP];
int prime[MAXP],mu[MAXP],cnt;
void Mu(){
    vis[1]=true,mu[1]=1;
    _for(i,2,MAXP){
        if(!vis[i])mu[i]=-1,prime[cnt++]=i;
        for(int j=0;j<cnt&&i*prime[j]<MAXP;j++){
            vis[i*prime[j]]=true;
            if(i%prime[j])
                mu[i*prime[j]]=-mu[i];
            else{
                mu[i*prime[j]]=0;
                break;
            }
        }
    }
}
```

```

}

int Ceil(int x,int y){return x%y?x/y+1:x/y;}

LL cal(int i,int n,int m,int k){
    LL ans=0;
    int lef=1,rig=0,tlef=Ceil(left,k),trig;
    while(tlef*k<=i){
        rig=min(i,min(n/(n/left),m/(m/left)));
        trig=rig/k;
        if(trig>=tlef)
            ans+=1LL*(mu[trig]-mu[tlef-1])*(n/left)*(m/left);
        left=rig+1;
        tlef=Ceil(left,k);
    }
    return ans;
}

int main()
{
    Mu();
    for(i,2,MAXP)
        mu[i]+=mu[i-1];
    int t=read_int(),a,b,d;
    while(t--){
        a=read_int(),b=read_int(),d=read_int();
        enter(cal(min(a,b),a,b,d));
    }
    return 0;
}

```

题解 2

$$\text{Ans} = \sum_{i=1}^a \sum_{j=1}^b [(i,j) == d] = \sum_{d \mid i} \sum_{d \mid j} [(i,j) == d]$$

改为枚举 \$d\$ 的倍数，根据 \$(6)\$ 式有

$$\sum_{d \mid i} \sum_{d \mid j} [(i,j) == d] = \sum_i \left\lceil \frac{ad}{bd} \right\rceil \sum_j \left\lceil \frac{bd}{ad} \right\rceil [(i,j) == 1] = \sum_i \left\lceil \frac{ad}{bd} \right\rceil \sum_j \left\lceil \frac{bd}{ad} \right\rceil \sum_{k \mid (i,j)} \mu(k)$$

考虑改变枚举顺序，有

$$\sum_i \left\lceil \frac{ad}{bd} \right\rceil \sum_j \left\lceil \frac{bd}{ad} \right\rceil \sum_{k \mid (i,j)} \mu(k) = \sum_{k=1}^{\infty} \left\lceil \min \left(\left\lceil \frac{ad}{bd} \right\rceil, \left\lceil \frac{bd}{ad} \right\rceil \right) \right\rceil \sum_{i \leq \left\lfloor \frac{ad}{kd} \right\rfloor} \sum_{j \leq \left\lfloor \frac{bd}{ki} \right\rfloor} \mu(k) = \sum_{k=1}^{\infty} \left\lceil \min \left(\left\lceil \frac{ad}{bd} \right\rceil, \left\lceil \frac{bd}{ad} \right\rceil \right) \right\rceil \left\lceil \frac{ad}{kd} \right\rceil \left\lceil \frac{bd}{ki} \right\rceil \mu(k)$$

剩下部分数论分块即可，时间复杂度 \$O(\sqrt{n})\$

```
#include <cstdio>
```

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <cstring>
#include <cctype>
#include <queue>
#include <cmath>
#define _for(i,a,b) for(int i=(a);i<(b);++i)
#define _rep(i,a,b) for(int i=(a);i<=(b);++i)
#define mem(a,b) memset((a),(b),sizeof(a))
using namespace std;
typedef long long LL;
inline int read_int(){
    int t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline LL read_LL(){
    LL t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline void write(LL x){
    register char c[21],len=0;
    if(!x) return putchar('0'),void();
    if(x<0)x=-x,putchar('-');
    while(x)c[++len]=x%10,x/=10;
    while(len)putchar(c[len--]+48);
}
inline void space(LL x){write(x),putchar(' ');}
inline void enter(LL x){write(x),putchar('\n');}
const int MAXP=5e4+5;
bool vis[MAXP];
int prime[MAXP],mu[MAXP],cnt;
void Mu(){
    vis[1]=true,mu[1]=1;
    _for(i,2,MAXP){
        if(!vis[i])mu[i]=-1,prime[cnt++]=i;
        for(int j=0;j<cnt&&i*prime[j]<MAXP;j++){
            vis[i*prime[j]]=true;
            if(i%prime[j])
                mu[i*prime[j]]=-mu[i];
            else{
                mu[i*prime[j]]=0;
                break;
            }
        }
    }
}
```

```

}

LL cal(int i,int n,int m){
    LL ans=0;
    int lef=1,rig=0;
    while(lef<=i){
        rig=min(i,min(n/(n/lef),m/(m/lef)));
        ans+=1LL*(mu[rig]-mu[lef-1])*(n/lef)*(m/lef);
        lef=rig+1;
    }
    return ans;
}
int main()
{
    Mu();
    _for(i,2,MAXP)
    mu[i]+=mu[i-1];
    int t=read_int(),a,b,d;
    while(t--){
        a=read_int(),b=read_int(),d=read_int();
        a/=d,b/=d;
        enter(cal(min(a,b),a,b));
    }
    return 0;
}

```

比较题解 1 与题解 2 最后的式子，发现

$$\begin{aligned} & \sum_{k=1}^{\lfloor \frac{a}{d} \rfloor} \sum_{j=1}^{\lfloor \frac{b}{d} \rfloor} \tau(j) \\ &= \sum_{k=1}^{\lfloor \frac{ad}{d} \rfloor} \sum_{j=1}^{\lfloor \frac{bd}{d} \rfloor} \tau(j) \end{aligned}$$

习题2

[洛谷p3327](#)

题意

给定 \$n,m\$ 求 \$\sum_{i=1}^n \sum_{j=1}^m \tau(ij)\$

引理

$$\tau(ij) = \sum_{x \mid i} \sum_{y \mid j} [x,y] = \sum_{x \mid i} \sum_{y \mid j} \sum_{z \mid \frac{ij}{xy}} z$$

证明

设 \$ij=p_1^{\alpha_1} p_2^{\alpha_2} \dots p_k^{\alpha_k}, i=p_1^{\beta_1} p_2^{\beta_2} \dots p_l^{\beta_l}\$

$p_k^{\beta_k}$ 对每个 ij 的因子 $z = p_1^{z_1} p_2^{z_2} \dots p_k^{z_k}$ 构造如下映射

$$\begin{aligned} &\begin{pmatrix} z_1 & z_2 & \dots & z_k \end{pmatrix} \\ &\longmapsto \begin{pmatrix} x_1 & x_2 & \dots & x_k & y_1 & y_2 & \dots & y_k \end{pmatrix} \\ &\text{其中 } \left\{ \begin{array}{l} x_i = [z_i] \leq \beta_i \\ y_i = [z_i] \gt \beta_i \end{array} \right. \end{aligned}$$

容易验证这是个双射，所以枚举 z 等价于枚举 $x = p_1^{x_1} p_2^{x_2} \dots p_k^{x_k}, y = p_1^{y_1} p_2^{y_2} \dots p_k^{y_k}, x \mid i, y \mid j, (x, y) = 1$ 证毕。

$$\varphi(ij) = \sum_{x \mid i} \sum_{y \mid j} \frac{1}{(x, y)}$$

证明方法为构造如下映射

$$\begin{aligned} &\begin{pmatrix} z_1 & z_2 & \dots & z_k \end{pmatrix} \\ &\longmapsto \begin{pmatrix} x_1 & x_2 & \dots & x_k & y_1 & y_2 & \dots & y_k \end{pmatrix} \\ &\text{其中 } \left\{ \begin{array}{l} x_i = [z_i] \leq \beta_i - z_i \\ y_i = [z_i] \gt \beta_i - z_i \end{array} \right. \end{aligned}$$

容易验证这是个双射，且 $z = \frac{1}{(x, y)}$

题解

根据引理，同时改变枚举顺序

$$\sum_{i=1}^n \sum_{j=1}^m \tau(ij) = \sum_{i=1}^n \sum_{j=1}^m \sum_{x \mid i} \sum_{y \mid j} (x, y) = \sum_{x=1}^n \sum_{y=1}^m \lfloor \frac{nx}{x} \rfloor \lfloor \frac{ny}{y} \rfloor$$

根据 \$(6)\$ 式并再次改变枚举顺序有

$$\begin{aligned} &\sum_{x=1}^n \sum_{y=1}^m \lfloor \frac{nx}{x} \rfloor \lfloor \frac{ny}{y} \rfloor = \sum_{x=1}^n \sum_{y=1}^m \lfloor \frac{nx}{x} \rfloor \lfloor \frac{ny}{y} \rfloor = \sum_{d=1}^{\min(n, m)} \mu(d) \sum_{x \mid d} \sum_{y \mid d} \lfloor \frac{nx}{x} \rfloor \lfloor \frac{ny}{y} \rfloor \end{aligned}$$

改为枚举倍数，再根据 \$(10)\$ 式有

$$\begin{aligned} &\sum_{d=1}^{\min(n, m)} \mu(d) \sum_{x \mid d} \sum_{y \mid d} \lfloor \frac{nd}{x} \rfloor \lfloor \frac{md}{y} \rfloor = \sum_{d=1}^{\min(n, m)} \mu(d) \sum_{x \mid d} \sum_{y \mid d} \lfloor \frac{nd}{x} \rfloor \lfloor \frac{md}{y} \rfloor = \sum_{d=1}^{\min(n, m)} \mu(d) \sum_{x \mid d} \sum_{y \mid d} \lfloor \frac{nd}{x} \rfloor \lfloor \frac{md}{y} \rfloor \end{aligned}$$

设 $f(n) = \sum_{i=1}^n \lfloor \frac{n}{i} \rfloor$ 则

$$f(n) = \sum_{d=1}^{\min(n, m)} \mu(d) f(\lfloor \frac{nd}{d} \rfloor) f(\lfloor \frac{md}{d} \rfloor)$$

$O(\left(n\sqrt{n}\right))$ 时间预处理出 f 后可以 $O(\sqrt{n})$ 处理每个询问。

```
#include <iostream>
#include <cstdio>
#include <cstdlib>
#include <algorithm>
#include <string>
#include <sstream>
#include <cstring>
#include <cctype>
#include <cmath>
#include <vector>
#include <set>
#include <map>
#include <stack>
#include <queue>
#include <ctime>
#include <cassert>
#define _for(i,a,b) for(int i=(a);i<(b);++i)
#define _rep(i,a,b) for(int i=(a);i<=(b);++i)
#define mem(a,b) memset(a,b,sizeof(a))
using namespace std;
typedef long long LL;
inline int read_int(){
    int t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline LL read_LL(){
    LL t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline char get_char(){
    char c=getchar();
    while(c==' '||c=='\n'||c=='\r')c=getchar();
    return c;
}
inline void write(LL x){
    register char c[21],len=0;
    if(!x)return putchar('0'),void();
    if(x<0)x=-x,putchar('-');
    while(x)c[++len]=x%10,x/=10;
    while(len)putchar(c[len--]+48);
}
inline void space(LL x){write(x),putchar(' ');}
inline void enter(LL x){write(x),putchar('\n');}
const int MAXP=5e4+5;
```

```
bool vis[MAXP];
int prime[MAXP], mu[MAXP], cnt, f[MAXP];
void Mu(){
    vis[1]=true, mu[1]=1;
    for(i,2,MAXP){
        if(!vis[i])mu[i]=-1, prime[cnt++]=i;
        for(int j=0;j<cnt&&i*prime[j]<MAXP;j++){
            vis[i*prime[j]]=true;
            if(i%prime[j])
                mu[i*prime[j]]=-mu[i];
            else{
                mu[i*prime[j]]=0;
                break;
            }
        }
    }
}
int cal(int n){
    int lef=1, rig=0, ans=0;
    while(left<=n){
        rig=n/(n/left);
        ans+=(rig-left+1)*(n/left);
        left=rig+1;
    }
    return ans;
}
inline LL cal(int i,int n,int m){
    int lef=1, rig=0;
    LL ans=0;
    while(left<=i){
        rig=min(n/(n/left),m/(m/left));
        ans+=1LL*f[n/left]*f[m/left]*(mu[rig]-mu[left-1]);
        left=rig+1;
    }
    return ans;
}
int main()
{
    Mu();
    for(i,2,MAXP)
        mu[i]+=mu[i-1];
    for(i,1,MAXP)
        f[i]=cal(i);
    int t=read_int(),n,m;
    while(t--){
        n=read_int(),m=read_int();
        enter(cal(min(n,m),n,m));
    }
    return 0;
}
```

{}

From:

<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:

https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:%E6%95%B0%E8%AE%BA_2&rev=1594302619

Last update: 2020/07/09 21:50

