数论 2

前置公式

\begin{equation}\left(\sum_{d\mid n}f(x)\right)\left(\sum_{d\mid m}g(x)\right)=\sum_{d_1\mid n}\sum_{d_2\mid m}f(d_1)g(d_2)\tag{1}\end{equation}

\begin{equation}\sum_{m=1}^n\sum_{d\mid (n,m)}f(d)=\sum_{d\mid n}\frac{f(d)n}d\tag{2}\end{equation}

\begin{equation}\sum_{x\mid n}\left(f(x)\sum_{y\mid\frac nx}g(y)\right)=\sum_{y\mid n}\left(g(y)\sum_{x\mid\frac ny}f(x)\right)\tag{3}\end{equation}

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

$(a)\ \exists n\left(\theta(n)\ne 0\right)$

$(b)\ \forall n\forall m \left((n,m)=1\to \theta(n\ast m)= \theta(n)\ast\theta(m)\right)$

性质

假定 θ 为可积函数，则

$$ \begin{array}{l} (1)\ \theta(1)=1\\ (2)\ \theta(n)=\theta\left(p_1^{\alpha_1}\right)\theta\left(p_2^{\alpha_2}\right)\ldots\theta\left(p_k^{\alpha_k}\right)\\ (3)\theta_1,\theta_2\text{可积}\to \theta_1\theta_2\text{可积}\\ \end{array} $$

定理

若 θ 为可积函数，则

$$ \begin{array}{l} (1)\ \psi(n)=\sum_{d\mid n}\theta(d)\text{可积}\\ (2)\ \psi(n)=\prod_{i=1}^k\left(1+\theta\left(p_k\right)+\theta\left(p_k^2\right)+\ldots+\theta\left(p_k^{\alpha_k}\right) \right)\end{array} $$

证明

设 $(n,m)=1$□则对 $d\mid nm$□一定存在唯一 $d_1\mid n,d_2\mid m,d=d_1d2$□

同时，对每对 $d_1\mid n,d_2\mid m$□一定有 $d=d_1d_2□d\mid nm$□

所以根据 (1) 式

\begin{equation}\psi(nm)=\sum_{d\mid nm} \theta(d)=\sum_{d_1\mid n}\sum_{d_2\mid m}\theta(d_1)\theta(d_2)=\left(\sum_{d\mid n}\theta(d)\right)\left(\sum_{d\mid m}\theta(d)\right)=\psi(n)\psi(m)\end{equation}

\begin{equation}\psi(n)=\prod_{i=1}^k\psi\left(p_k^{\alpha_k}\right)=\prod_{i=1}^k\left(1+\theta\left(p_k\right)+\theta\left(p_k^2\right)+\ldots+\theta\left(p_k^{\alpha_k}\right)\right)\tag{4}\end{equation}

莫比乌斯函数

定义

$$ \mu (n)= \begin{cases} 1 &n=1\\ (-1)^k &n=p_1p_2\ldots p_k\\ 0 &otherwise \end{cases} $$

性质

若 θ 为可积函数，根据 (4) 式

\begin{equation}\sum_{d\mid n}\mu (d)\theta (d)=\prod_{i=1}^k\left(1+\mu\left(p_k\right)\theta\left(p_k\right)+\mu\left(p_k^2\right)\theta\left(p_k^2\right)+\ldots+\mu\left(p_k^{\alpha_k}\right)\theta\left(p_k^{\alpha_k}\right)\right)=\prod_{i=1}^k\left(1-\theta\left(p_k\right)\right)\tag{5}\end{equation}

特别地

\begin{equation}\theta (n)\equiv 1,\sum_{d\mid n}\mu (d)=\begin{cases} 1 &n=1\\ 0 &otherwise\\ \end{cases}\tag{6}\end{equation}

\begin{equation}\theta (n)=\frac 1n,\sum_{d\mid n}\frac{\mu (d)}d=(1-p_1)(1-p_2)\ldots(1-p_k)\tag{7}\end{equation}

欧拉函数

定义

\begin{equation}\varphi(n)=\sum_{m=1}^n [(n,m)==1]\end{equation}

性质

根据 (2) 式、(6) 式和 (7) 式

\begin{equation}\varphi(n)=\sum_{m=1}^n [(n,m)=1]=\sum_{m=1}^n\sum_{d\mid (n,m)}\mu (d)=n\sum_{d\mid n}\frac {\mu (d)}d=n(1-p_1)(1-p_2)\ldots(1-p_k)\end{equation}

其他常见可积函数

\begin{equation}\tau(n)=\sum_{d\mid n}1=\prod_{i=1}^k(\alpha_k+1)\end{equation}

\begin{equation}\sigma (n)=\sum_{d\mid n}d=\prod_{i=1}^k\left(1+p_k+p_k^2+\ldots+p_k^{\alpha_k} \right)\end{equation}

完全积性函数

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

\begin{equation}e(n)=[n==1]\end{equation}

\begin{equation}I(n)=1\end{equation}

\begin{equation}id(n)=n\end{equation}

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

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，g 的迪利克雷卷积运算为 \begin{equation}(f\ast g)(n)=\sum_{d\mid n}f(d)\ast g(\frac nd)\end{equation}

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 $f(1)\ne 0$ 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\begin{equation}g(n)=\frac 1{f(1)}\left(e(n)-\sum_{d\mid n,d\ne 1}f(d)\ast g(\frac nd)\right)\end{equation}

知 $g(n)$ 可由 $g(1\sim n-1)$ 递推得到，且 \begin{equation}\sum_{d\mid n}f(d)\ast g(\frac nd)=f(1)\ast g(n)+\sum_{d\mid n,d\ne 1}f(d)\ast g(\frac nd)=e(n)\end{equation}

莫比乌斯反演定理

设 $F(x)$，$f(x)$ 为数论函数 \begin{equation}F(n)=\sum_{d\mid n}f(d)\iff f(n)=\sum_{d\mid n}\mu(d)F(\frac nd)\tag{8}\end{equation}

该定理的另一种形式为 \begin{equation}F(n)=\sum_{n\mid d}f(d)\iff f(n)=\sum_{n\mid d}\mu (\frac dn)F(d)\tag{9}\end{equation}

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

证明

根据 (3) 式和 (6) 式

\begin{equation}\sum_{d\mid n}\mu (d)F(\frac nd)=\sum_{d\mid n}\left(\mu (d)\sum_{d_1\mid \frac nd}f(d_1)\right)=\sum_{d_1\mid n}\left(f(d_1)\sum_{d\mid \frac n{d_1}}\mu (d)\right)=f(n)\end{equation}

一些常见的狄利克雷卷积

1$I\ast \mu=e$

2$\mu\ast id=\varphi$

3$I\ast id=\sigma$

4$I\ast I=\tau$

5$I\ast \varphi=id$

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□$I\ast \varphi=I\ast (\mu\ast id)=(I\ast \mu)\ast id=e\ast id=id$□

证明 6

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

算法练习

习题1

洛谷p3455

题意

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

题解 1

在约束条件 $i\le a,j\le b$ 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{equation}f(n)=\sum_{i=1}^a\sum_{j=1}^b[(i,j)==n]\end{equation}

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

根据莫比乌斯反演定理，有

\begin{equation}Ans=f(d)=\sum_{d\mid k}^{k\le min(a,b)}\mu (\lfloor \frac kd\rfloor)F(k)=\sum_{d\mid k}^{k\le min(a,b)}\mu (\lfloor \frac kd\rfloor)\lfloor\frac ak\rfloor\lfloor \frac bk\rfloor\end{equation}

设 $k=t\ast d$□改为枚举 t□有

\begin{equation}\sum_{d\mid k}^{k\le min(a,b)}\mu (\lfloor \frac kd\rfloor)\lfloor\frac ak\rfloor\lfloor \frac bk\rfloor=\sum_{t=1}^{t\ast d\le \min(a,b)}\mu (t)\lfloor\frac a{t\ast d}\rfloor\lfloor \frac b{t\ast d}\rfloor\end{equation}

剩下部分数论分块即可，时间复杂度 $O\left(\max\left(\sqrt a,\sqrt b\right)\right)$□

```
#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(lef,k),trig;
    while(tlef*k<=i){
        rig=min(i,min(n/(n/lef),m/(m/lef)));
        trig=rig/k;
        if(trig>=tlef)
        ans+=1LL*(mu[trig]-mu[tlef-1])*(n/lef)*(m/lef);
        lef=rig+1;
        tlef=Ceil(lef,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

\begin{equation}Ans=\sum_{i=1}^a\sum_{j=1}^b[(i,j)==d]=\sum_{d\mid i}\sum_{d\mid j}[(i,j)==d]\end{equation}

改为枚举 d 的倍数，根据 (6) 式有

\begin{equation}\sum_{d\mid i}\sum_{d\mid j}[(i,j)==d]=\sum_i^{\lfloor \frac ad\rfloor}\sum_j^{\lfloor \frac bd\rfloor}[(i,j)==1]=\sum_i^{\lfloor \frac ad\rfloor}\sum_j^{\lfloor \frac bd\rfloor}\sum_{k\mid(i,j)}\mu(k)\end{equation}

考虑改变枚举顺序，有

\begin{equation}\sum_i^{\lfloor\frac ad\rfloor}\sum_j^{\lfloor\frac bd\rfloor}\sum_{k\mid(i,j)}\mu(k)=\sum_{k=1}^{\min(\lfloor\frac ad\rfloor,\lfloor\frac bd\rfloor)}\sum_{k\mid i}^{i\le\lfloor\frac ad\rfloor}\sum_{k\mid j}^{j\le\lfloor\frac bd\rfloor}\mu(k)=\sum_{k=1}^{\min(\lfloor\frac ad\rfloor,\lfloor\frac bd\rfloor)}\lfloor\frac{\lfloor\frac ad\rfloor}k\rfloor\lfloor\frac{\lfloor\frac bd\rfloor}k\rfloor\mu(k)\end{equation}

剩下部分数论分块即可，时间复杂度 $O\left(\max\left(\sqrt a,\sqrt b\right)\right)$。

```
#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{equation}\sum_{k=1}^{k\ast d\le \min(a,b)}\lfloor\frac a{k\ast d}\rfloor\lfloor \frac b{k\ast d}\rfloor=\sum_{k=1}^{\min(\lfloor\frac ad\rfloor,\lfloor\frac bd\rfloor)}\lfloor\frac{\lfloor\frac ad\rfloor}k\rfloor\lfloor\frac{\lfloor\frac bd\rfloor}k\rfloor\tag{10}\end{equation}

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

\begin{equation}\tau(ij)=\sum_{x\mid i}\sum_{y\mid j}[(x,y)==1]\tag{11}\end{equation}

证明

设 $ij=p_1^{\alpha_1}p_2^{\alpha_2}\ldots p_k^{\alpha_k},i=p_1^{\beta_1}p_2^{\beta_2}\ldots p_k^{\beta_k}$□对每个 ij 的因子 $z=p_1^{z_1}p_2^{z_2}\ldots p_k^{z_k}$□构造如下映射

\begin{equation}\begin{pmatrix}z_1&z_2&\cdots &z_k\\

\end{pmatrix}\longleftrightarrow\begin{pmatrix}x_1&x_2&\cdots &x_k\\y_1&y_2&\cdots &y_k\\ \end{pmatrix},\text{其中}\left \{ \begin{array}{l} x_i=[z_i\le \beta_i]z_i \\ y_i=[z_i\gt \beta_i](z_i-\beta_i) \end{array} \right.\end{equation}

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

顺便给出式子\begin{equation}\varphi(ij)=\sum_{x\mid i}\sum_{y\mid j}\frac{iy}x[(x,y)==1]\tag{12}\end{equation}

证明方法为构造如下映射

\begin{equation}\begin{pmatrix}z_1&z_2&\cdots &z_k\\ \end{pmatrix}\longleftrightarrow\begin{pmatrix}x_1&x_2&\cdots &x_k\\y_1&y_2&\cdots &y_k\\ \end{pmatrix},\text{其中}\left \{ \begin{array}{l} x_i=[z_i\le \beta_i](\beta_i-z_i) \\ y_i=[z_i\gt \beta_i](z_i-\beta_i) \end{array} \right.\end{equation}

容易验证这是个双射，且 $z=\frac{iy}x$□

题解

根据引理，同时改变枚举顺序有\begin{equation}\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)==1]=\sum_{x=1}^n\sum_{y=1}^m\lfloor\frac nx\rfloor\lfloor\frac my\rfloor[(x,y)==1]\end{equation}

根据 (6) 式并再次改变枚举顺序有

\begin{equation}\sum_{x=1}^n\sum_{y=1}^m\lfloor\frac nx\rfloor\lfloor\frac my\rfloor[(x,y)==1]=\sum_{x=1}^n\sum_{y=1}^m\lfloor\frac nx\rfloor\lfloor\frac my\rfloor\sum_{d=1}^{(x,y)}\mu(d)=\sum_{d=1}^{\min(n,m)}\sum_{d\mid x}^{x\le n}\sum_{d\mid y}^{y\le m}\lfloor\frac nx\rfloor\lfloor\frac my\rfloor\mu(d)\end{equation}

改为枚举倍数，再根据 (10) 式有

\begin{equation}\sum_{d=1}^{\min(n,m)}\sum_{d\mid x}^{x\le n}\sum_{d\mid y}^{y\le m}\lfloor\frac nx\rfloor\lfloor\frac my\rfloor\mu(d)=\sum_{d=1}^{\min(n,m)}\sum_{x=1}^{\lfloor\frac nd\rfloor}\sum_{y=1}^{\lfloor\frac md\rfloor}\lfloor\frac{\lfloor\frac nd\rfloor}x\rfloor\lfloor\frac{\lfloor\frac md\rfloor}y\rfloor\mu(d)=\sum_{d=1}^{\min(n,m)}\mu(d)\left(\sum_{x=1}^{\lfloor\frac nd\rfloor}\lfloor\frac{\lfloor\frac nd\rfloor}x\rfloor\right)\left(\sum_{y=1}^{\lfloor\frac md\rfloor}\lfloor\frac{\lfloor\frac md\rfloor}y\rfloor\right)\end{equation}

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

\begin{equation}\text{Ans}=\sum_{d=1}^{\min(n,m)}\mu(d)f(\lfloor\frac nd\rfloor)f(\lfloor\frac md\rfloor)\end{equation}

$O\left(n\sqrt n\right)$ 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(lef<=n){
        rig=n/(n/lef);
        ans+=(rig-lef+1)*(n/lef);
        lef=rig+1;
    }
    return ans;
}
inline LL cal(int i,int n,int m){
    int lef=1,rig=0;
    LL ans=0;
    while(lef<=i){
        rig=min(n/(n/lef),m/(m/lef));
        ans+=1LL*f[n/lef]*f[m/lef]*(mu[rig]-mu[lef-1]);
        lef=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;
}
```