# 莫比乌斯反演

wcz[1]

January 7, 2018

[1]Contact me:aiyoupass@outlook.com

# Contents

# 1 前言

本文内容大部分来自 Oier PoPoQQQ 的课件。
Download: onedrive, baidu pan, 密码:6ug5

本文基本上由我学习相当于是制作的一篇学习笔记，但是将课件中的一些不完善的地方加以完善使得更容易理解，加上了部分例题的代码

阅读本文章需要的前置知识:

- ✏ 积性函数
- ✏ 组合数学
- ✏ 线性筛法
- ✏ 容斥原理
- ✏ 狄利克雷卷积

事实上莫比乌斯反演跟积性函数、欧拉函数有着密切的联系。

## 2 引子

介绍莫比乌斯反演之前我们先来看一个函数

$$F(n) = \sum_{d|n} f(d) \tag{1}$$

根据 $F(n)$ 的定义

$$\begin{aligned}
&F(1) = f(1)\\
&F(2) = f(1) + f(2)\\
&F(3) = f(1) + f(3)\\
&F(4) = f(1) + f(2) + f(4)\\
&F(5) = f(1) + f(5)\\
&F(6) = f(1) + f(2) + f(3) + f(6)\\
&F(7) = f(1) + f(7)\\
&F(8) = f(1) + f(2) + f(4) + f(8)\\
&\cdots
\end{aligned}$$

于是我们便可以通过 $F(n)$ 推导出 $f(n)$

$$\begin{aligned}
&f(1) = F(1)\\
&f(2) = F(2) - F(1)\\
&f(3) = F(3) - F(1)\\
&f(4) = F(4) - F(2)\\
&f(5) = F(5) - F(1)\\
&f(6) = F(6) - F(3) - F(2) + F(1)\\
&f(7) = F(7) - F(1)\\
&f(8) = F(8) - F(4)\\
&\cdots
\end{aligned}$$

在推导的过程中我们是否发现了一些规律?

# 3 莫比乌斯反演

莫比乌斯反演[1]

## 3.1 莫比乌斯反演定义

$$F(n) = \sum_{d|n} f(n) = \sum_{d|n} \mu(d) f(\frac{n}{d}) \tag{2}$$

其中 $\mu(d)$ 为莫比乌斯函数，定义如下

$$\mu(d) = \begin{cases} 1, & d = 0 \\ (-1)^k, & d = p_1 p_2 \cdots p_k, \forall p_i != p_j \\ 0, & others \end{cases} \tag{3}$$

莫比乌斯函数的定义式[2]

[1]baike: 莫比乌斯反演是数论数学中很重要的内容，可以用于解决很多组合数学的问题。
[2]$\mu(n) = \delta_{\varpi(n)\Omega(n)} \lambda(n)$

### 3.2 莫比乌斯函数的性质

**(1)**

当 n 不等于 1 时，n 所有因子的莫比乌斯函数值的和为 0，
这个式子可以写成这种形式:

$$\sum_{d|n}\mu(d)=[n=1] \tag{4}$$

事实上就是由这个式子决定了莫比乌斯函数

设

$$F(n)=\sum_{d|n}\mu(d)$$

证明:

①当 $n=1$ 的时候显然成立

②当 $n\neq 1$ 时, $n=p_1^{a_1}p_2^{a_2}\cdots p_k^{a_k}$

$\because \mu(d)\neq 0 \Leftrightarrow d=p_1p_2\cdots p_t$

质因子个数为 $r$ 的因子只有 $C_k^r$ 个

$\therefore F(x)=C_k^0-C_k^1+C_k^2+\cdots+(-1)^kC_k^k=\sum\limits_{i=0}^{k}(-1)^iC_k^i$

接下来只需证明 $\sum\limits_{i=0}^{n}(-1)^iC_n^i=0(n\in N_+)$ 即可

因为二项式定理

$(x+y)^n=\sum\limits_{i=0}^{n}C_n^ix^iy^{n-i}$

令 $x=-1,y=1,$ 代入即可得证。

**(2)**

对于 $n\in N_+$ 有:

$$\sum_{d|n}\frac{\mu(d)}{d}=\frac{\phi(n)}{n} \tag{5}$$

$$\text{只需要令}F(n)=n,f(n)=\phi(n)$$
$$\text{代入}F(n)=\sum_{d|n}f(n)=\sum_{d|n}\mu(d)f(\frac{n}{d})\text{即可}$$
$$\text{那么就有}n=\sum_{d|n}\phi(d)\text{为什么成立?}$$

**(3)**

积性函数数论上积性函数的定义

设函数 $f(n)$, 其中 $n\in N+$
对于任意 $(x,y)=1$ 都有 $f(xy)=f(x)f(y)$,
则称 $f(n)$ 为一个积性函数;
若对于任意 $x,y$ 都有 $f(xy)=f(x)f(y)$,
则称 $f(n)$ 为一个完全积性函数。

积性函数的性质

①$f(1)=1$
②积性函数的前缀和也是积性函数

通过线性筛求出莫比乌斯函数的值

```
mu[1]=1;
for(i=2;i<=n;i++){
        if(!not_prime[i]){
                prime[++tot]=i;
                mu[i]=-1;
        }
        for(j=1;prime[j]*i<=n;j++){
                not_prime[prime[j]*i]=1;
                if(i%prime[j]==0){
                        mu[prime[j]*i]=0;
                        break;
                }
                mu[prime[j]*i]=-mu[i];
        }
}
```

### 3.2.1 例题 1:

BZOJ 2440 完全平方数

题目大意：求第 k 个无平方因子数[3]

做法：首先二分答案，问题转化为求 $[1,x]$ 之间有多少个无平方因子数根据容斥原理可知，对于 $sqrt(x)$ 之内所有的质数，答案 G(x)=0 个质数平方倍数的个数 -1 个质数平方倍数的个数 +2 个质数平方倍数的个数 -...，那么对于偶数个质数平方对于答案的贡献就是正的，否则是负的，
如果不是若干个互异质数的乘积，那么对答案没有影响，

如何表示这个式子呢?

观察莫比乌斯函数的定义3.1，可以知道对于能对答案产生贡献的数 $x$，$\mu(x)=(-1)^k$，其中 $k$ 为 $x$ 分解得到质数的个数根据上述说明，那么可以得知结果

$$G(x)=\sum_{i=1}^{\lfloor\sqrt{x}\rfloor}\mu(i)\lfloor\frac{x}{i^2}\rfloor \tag{6}$$

```
#include<iostream>
#include<cstdio>
#include<cmath>
#define N 100005
using namespace std;

bool not_prime[N];
int prime[N];
int mu[N];
int tot;

void Mu(int n){
    int i,j;
    mu[1]=1;
    for(i=2;i<=n;i++){
        if(!not_prime[i]){
                prime[++tot]=i;
                mu[i]=-1;
        }
```

[3]无平方因子数 (Square-Free Number)，即分解之后所有质因数的次数都为 1 的数

```
            for(j=1;prime[j]*i<=n;j++){
                    not_prime[prime[j]*i]=1;
            if(i%prime[j]==0){
                        mu[prime[j]*i]=0;
                        break;
                }
                    mu[prime[j]*i]=-mu[i];
            }
    }
}
int can(int x){
    int sum=0;
    int s=floor(sqrt(x));
    for(int i=1;i<=s;++i)
        if(mu[i])
            sum+=mu[i]*floor(x/(i*i));
    return sum;
}

int main(){
    Mu(N); int T,sum;
    scanf("%d",&T);
    while(T--){
        scanf("%d",&num);
        long long l=1,r=num<<1,mid;
        while(l<r){
            mid=(l+r)>>1;
            if(can(mid)<num)
                l=mid+1;
            else r=mid;
        }
        printf("%lld\n",r);
    }
    return 0;
}
```

### 3.3 莫比乌斯反演定理的证明

$$F(n)=\sum_{d|n}f(d)\Rightarrow f(n)=\sum_{d|n}\mu(d)F(\frac{n}{d}) \tag{7}$$

证明:

$$\begin{aligned}f(n)&=\sum_{d|n}\mu(d)F(\frac{n}{d})\\&=\sum_{d|n}\mu(d)\sum_{k|\frac{n}{d}}f(k)\\&=\sum_{k|n}f(k)\sum_{d|\frac{n}{k}}\mu(d)\\&=f(n)\end{aligned}$$

形式二:

$$F(n)=\sum_{n|d}f(d)\Rightarrow f(n)=\sum_{n|d}\mu(\frac{d}{n})F(d) \tag{8}$$

证明同理，一般要用到的都是这种形式

### 3.4 莫比乌斯反演的应用

主要是用于处理一些组合数问题。

对于一些函数 $f(n)$, 如果我们很难直接求出它的值，而容易求出倍数和或约数和 $F(n)$, 那么我们可以直接利用莫比乌斯反演来求得 $f(n)$ 的值。

例: f(n) 表示某一范围内 (x,y)=n 的数对的数量，
F(n) 表示某一范围内 n|(x,y) 的数对的数量
那么直接求 f(n) 并不是很好求，而 F(n) 求起来相对无脑一些，
那么我们可以通过对 F(n) 进行莫比乌斯反演来求得 f(n)。

#### 3.4.1 例题 2:

BZOJ 2301 Problem b

题目大意：询问有多少对 $(x,y)$ 满足 $x\in[a,b]$ , $y\in[c,d]$ 且 $(x,y)=k$。

根据容斥原理，这个题目就可以转化成

$$s_1 = \sum_{i=1}^{b}\sum_{j=1}^{d}[(i,j)=k]$$
$$s_2 = \sum_{i=1}^{a-1}\sum_{j=1}^{d}[(i,j)=k]$$
$$s_3 = \sum_{i=1}^{b}\sum_{j=1}^{c-1}[(i,j)=k]$$
$$s_4 = \sum_{i=1}^{a-1}\sum_{j=1}^{c-1}[(i,j)=k]$$

其中答案为 $s_1 - s_2 - s_3 + s_4$

那么我们需要快速求出

$$\sum_{i=1}^{a}\sum_{j=1}^{b}[(i,j)=k] \tag{9}$$

这个式子可以进一步转化为

$$\sum_{i=1}^{\lfloor\frac{a}{k}\rfloor}\sum_{j=1}^{\lfloor\frac{b}{k}\rfloor}[(i,j)=k] \tag{10}$$

考虑莫比乌斯反演, 令

$$f(k) = \sum_{i=1}^{a}\sum_{j=1}^{b}[(i,j)=k] \tag{11}$$
$$F(k) = \sum_{i=1}^{a}\sum_{j=1}^{b}[k|(i,j)] \tag{12}$$
$$=\lfloor\frac{a}{k}\rfloor\lfloor\frac{b}{k}\rfloor \tag{13}$$
$$\text{反演后可得}f(k) = \sum_{k|d}\mu(\frac{d}{k})F(d) \tag{14}$$
$$= \sum_{k|d}\mu(\frac{d}{k})\lfloor\frac{a}{d}\rfloor\lfloor\frac{b}{d}\rfloor \tag{15}$$

分析可知这个算法的复杂度是 $\Theta(n)$

我们还需要对这个算法进行进一步优化

因为 $\lfloor\frac{a}{d}\rfloor$ 至多只有 $2\sqrt{a}$ 个取值,

那么 $\lfloor\frac{a}{d}\rfloor\lfloor\frac{b}{d}\rfloor$ 至多只有 $2(\sqrt{a}+\sqrt{b})$ 个取值

因为使得 $\lfloor\frac{a}{d}\rfloor\lfloor\frac{b}{d}\rfloor == i$ 成立的 $i$ 值都是连续的,

所以可以维护一个莫比乌斯函数的前缀和,

这样就可以在 $\Theta(\sqrt{n})$ 的时间内出解

枚举除法的取值在莫比乌斯反演的应用当中非常常用

```
if(a>b)swap(a,b);
for(i=1;i<=a;i=last+1){
        last=min(a/(a/i),b/(b/i));
        re+=(a/i)*(a/i)*(sum[last]-sum[i-1]);
}
return re;
```

代码异常好写

```
#include<iostream>
#include<cstdio>
#include<cmath>
#define N 50005
#define inf 0x7ffffff
using namespace std;

bool not_prime[N];
int prime[N];
int sum[N];
int mu[N];
int tot;

void Mu(int n){
    int i,j;
    mu[1]=1;
    for(i=2;i<=n;i++){
        if(!not_prime[i]){
            prime[++tot]=i;
            mu[i]=-1;
        }
        for(j=1;prime[j]*i<=n;j++){
            not_prime[prime[j]*i]=1;
            if(i%prime[j]==0){
                mu[prime[j]*i]=0;
                break;
            }
            mu[prime[j]*i]=-mu[i];
        }
    }
    for(int i=1;i<=n;++i)
        sum[i]=sum[i-1]+mu[i];
}
int ans(int n,int m){
    if(n>m)swap(n,m);
    int last,i,re=0;
    for(i=1;i<=n;i=last+1){
        last=min(n/(n/i),m/(m/i));
```

```
            re+=(n/i)*(m/i)*(sum[last]-sum[i-1]);
    }
    return re;
}

int main(){
    Mu(N);
    int T;
    int a,b,c,d,k;
    scanf("%d",&T);
    while(T--){
        scanf("%d%d%d%d%d",&a,&b,&c,&d,&k);
        a--;c--;
        a/=k;b/=k;c/=k;d/=k;
        int Ans=ans(b,d)-ans(a,d)-ans(b,c)+ans(a,c);
        printf("%d\n",Ans);
    }
    return 0;
}
```

BZOJ 10s 但是 luogu 却莫名 WA
全部加上 long long 之后总算是过了
百思不得其解

### 3.4.2 例题 3

BZOJ 2820 YY 的 GCD

题目大意：求有多少数对 $(x, y)$ 满足 $x \in [1, n], y \in [1, m]$ 满足 $(x, y)$ 为质数

做法：

首先这个题目和上一个题目不一样的地方是他需要一个特殊的转化

$$\text{令} k = min(n, m); \tag{16}$$

$$ans = \sum_{p}^{k} \sum_{i=1}^{n} \sum_{j=1}^{m} [(i, j) = p] \tag{17}$$

$$= \sum_{p}^{k} \sum_{d=1}^{k} \mu(d) \lfloor \frac{n}{pd} \rfloor \lfloor \frac{m}{pd} \rfloor \tag{18}$$

$$\text{令} T = pd \tag{19}$$

$$ans = \sum_{T=1}^{k} \lfloor \frac{n}{T} \rfloor \lfloor \frac{m}{T} \rfloor \sum_{p|T}^{k} \mu(\frac{T}{p}) \tag{20}$$

$$\text{令} F(k) = \sum_{p|T}^{k} \mu(\frac{T}{p}) \tag{21}$$

$$\text{则} ans = \sum_{T=1}^{k} F(k) \lfloor \frac{n}{T} \rfloor \lfloor \frac{m}{T} \rfloor \tag{22}$$

$$\tag{23}$$

线性筛素数的时候对 $F(k)$ 前缀和处理

然后就转变为和例二3.4.1一样的做法，枚举除法的取值了

```
#include<iostream>
#include<cstdio>
#include<cmath>
#define N 10000000
#define ll long long
using namespace std;

bool not_prime[N];
ll prime[N];
ll sum[N];
ll mu[N];
```

```
ll tot;

void Mu(int n){
    mu[1]=1;
    for(int i=2;i<=n;i++){
        if(!not_prime[i]){
            prime[++tot]=i;
            mu[i]=-1;
        }
        for(int j=1;prime[j]*i<=n;j++){
            not_prime[prime[j]*i]=1;
            if(i%prime[j]==0){
                mu[prime[j]*i]=0;
                break;
            }
            mu[prime[j]*i]=-mu[i];
        }
    }
    for(int i=1;i<=tot;++i)
        for(int j=1;j*prime[i]<=n;++j)
            sum[j*prime[i]]+=(ll)mu[j];
    for(int i=1;i<=n;++i)
        sum[i]+=(ll)sum[i-1];
}

ll ans(int n,int m){
    if(n>m)swap(n,m);
    int last,i;ll re=0;
    for(i=1;i<=n;i=last+1){
        last=min(n/(n/i),m/(m/i));
        re+=(ll)(n/i)*(m/i)*(sum[last]-sum[i-1]);
    }
    return re;
}

int main(){
    Mu(N);
    int T;
    int a,b;
    scanf("%d",&T);
```

```
    while(T--){
        scanf("%d%d",&a,&b);
        ll Ans=ans(a,b);
        printf("%lld\n",Ans);
    }
    return 0;
}
```

这已经是第 n 次被 long long 卡一个小时以上了

#### 3.4.3 例题 4:

BZOJ 3529: [Sdoi2014] 数表
选择膜拜 Po 爷

题目大意：令 F(i) 为 i 的约数和，给定 n,m,a, 求

$$\sum_{F(d)\le a} F((i,j)=d)$$

因为 a 的限制非常讨厌，所以我们先忽略它的存在
令 $Z=min(n,m)$

$$\begin{aligned}
\text{令}g(i) &= \sum[(x,y)=i] \\
\text{那么显然有}g(i) &= \sum_{i|d}\mu(\frac{d}{i})\lfloor\frac{n}{d}\rfloor\lfloor\frac{m}{d}\rfloor \\
\because (i,j) =& d, d=p_1^{a_1}p_2^{a_2}\cdots p_k^{a_k} \\
\text{由约数和定理得}F(d) =& (p_1^0+p_1^1+\cdots+p_1^{a_1})(p_2^0+p_2^1+\cdots+p_2^{a_2})\cdots(p_k^0+p_k^1+\cdots p_k^{a_k}) \\
\therefore F(pq) =& F(p)F(q),(p,q)=1 \\
& F(i)\text{ 可以利用线性筛在 }O(n)\text{ 时间内处理出来，那么就有} \\
ans &= \sum_{i=1}^{Z}F(i)g(i) \\
&= \sum_{d=1}^{Z}\lfloor\frac{n}{d}\rfloor\lfloor\frac{m}{d}\rfloor\sum_{i|d}F(i)\mu(\frac{d}{i})
\end{aligned}$$

然后我们可以发现最后这个式子的形式和上面非常像
然后利用上述前缀和和枚举除法取值的方法就可以完成

可是题目中还有一个 $a$ 的限制，我们可以发现对答案有贡献的只有 $F(i)\le ai$

可以离线来处理这个东西，将询问按照 $a$ 从小到大排序，将 $F(i)$ 按照从小到大的顺序排序，每次询问之前将所有 $F(i)\le a$ 插入并且用树状数组维护前缀和。当然还有一个需要注意的问题是取模[4]
接连做了好几道题都有喜闻乐见的除法分块3.4.1

[4]取模可以利用自然溢出 $int$ 最后再对 $2^{31}-1$ 取与即可

```
#include<algorithm>
#include<iostream>
#include<cstdio>
#define N 100005
#define inf 0x7ffffff
#define ll long long
using namespace std;
int read(){
    int x=0,f=1;char ch=getchar();
    while(ch<'0'||ch>'9'){if(ch=='-')f=-1;ch=getchar();}
    while(ch>='0'&&ch<='9'){x=x*10+ch-'0';ch=getchar();}
    return x*f;
}
int Q,mx,cnt;
struct data{
    int n,m,a,id;
}q[N];
bool operator<(data a,data b){
    return a.a<b.a;
}
pair<int,int>F[N];
int pri[N],mu[N];
int ans[N],t[N];
bool mark[N];
void add(int x,int val){
    for(int i=x;i<=mx;i+=i&-i)t[i]+=val;
}
int query(int x){
    int tmp=0;for(int i=x;i;i-=i&-i)tmp+=t[i];return tmp;
}
int Query(int l,int r){
    return query(r)-query(l-1);
}
void getmu(){
    mu[1]=1;
    for(int i=2;i<=mx;i++){
        if(!mark[i])pri[++cnt]=i,mu[i]=-1;
        for(int j=1;j<=cnt&&pri[j]*i<=mx;j++){
            mark[pri[j]*i]=1;
```

```
                if(i%pri[j]==0){mu[pri[j]*i]=0;break;}
                else mu[pri[j]*i]=-mu[i];
            }
        }
    for(int i=1;i<=mx;i++)
        for(int j=i;j<=mx;j+=i)
            F[j].first+=i;
    for(int i=1;i<=mx;i++)F[i].second=i;
}
void solve(int x){
    int id=q[x].id,n=q[x].n,m=q[x].m;
    for(int i=1,j;i<=q[x].n;i=j+1){
        j=min(n/(n/i),m/(m/i));
        ans[id]+=(n/i)*(m/i)*Query(i,j);
    }
}
int main(){
    Q=read();
    for(int i=1;i<=Q;i++){
        q[i].n=read();q[i].m=read();q[i].a=read();
        q[i].id=i;if(q[i].n>q[i].m)swap(q[i].n,q[i].m);
        mx=max(mx,q[i].n);
    }
    getmu();int now=0;
    sort(q+1,q+Q+1);sort(F+1,F+mx+1);
    for(int i=1;i<=Q;i++){
      while(now+1<=mx&&F[now+1].first<=q[i].a){
      for(int j=F[++now].second;j<=mx;j+=F[now].second)
                add(j,F[now].first*mu[j/F[now].second]);
        }solve(i);
    }
    for(int i=1;i<=Q;i++)
        printf("%d\n",ans[i]&inf);
    return 0;
}
```

### 3.4.4 例题 5:

BZOJ 2154:Crash 的数字表格, 据说和"luogu P3768"惊人的相似

题目大意: 给定 $n, m$，求

$$\sum_{i=1}^{n}\sum_{j=1}^{m}[i,j]$$

~~感觉有点慌，完整的公式推导花了两页~~

$$ans = \sum_{i=1}^{n}\sum_{j=1} m\,[i,j]$$

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

$$d = (i,j)$$

$$F(x,y) = \sum_{i=1}^{x}\sum_{j=1}^{y} ij\,[(i,j)=1]$$

$$ans = \sum_{d=1}^{n}\frac{d^2F(\frac{n}{d},\frac{m}{d})}{d}$$

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

$$F(x) = \sum_{i=1}^{x}\sum_{j=1}^{y} ij\sum_{d|(i,j)}\mu(d)$$

$$= \sum_{d=1}^{x}\mu(d)d^2\sum_{i=1}^{\frac{x}{d}} i\sum_{j=1}^{\frac{y}{d}} j$$

$$= \sum_{d=1}^{x}\mu(d)d^2\frac{\frac{x}{d}(\frac{x}{d}+1)}{2}\frac{\frac{y}{d}(\frac{y}{d}+1)}{2}$$

$$ans = \sum_{d=1}^{n} dF(\frac{n}{d},\frac{m}{d})$$

$$= \sum_{d=1}^{n} d\sum_{i=1}^{\frac{n}{d}}\mu(i)i^2\frac{\frac{n}{di}(\frac{n}{di}+1)}{2}\frac{\frac{m}{di}(\frac{m}{di}+1)}{2}$$

```
#include<cstdio>
#include<cstring>
#include<iostream>
#include<algorithm>
#define M 10001000
#define MOD 20101009
#define C 10050505
using namespace std;
typedef long long ll;

ll n,m,ans,sum[M];
int mu[M]={0,1},prime[M],tot;
bool not_prime[M];

void getmu(){
	int i,j;
	for(i=2;i<M;i++){
		if(!not_prime[i])
			prime[++tot]=i,mu[i]=-1;
		for(j=1;j<=tot&&prime[j]*i<M;j++){
			not_prime[prime[j]*i]=1;
			if(i%prime[j]==0){
				mu[prime[j]*i]=0;
				break;
			}
			mu[prime[j]*i]=-mu[i];
		}
	}
	for(i=1;i<M;i++)
	sum[i]=(sum[i-1]+(long long)i*i*mu[i])%MOD;
}
inline ll S(ll x){
	return (x%MOD)*(x+1%MOD)%MOD*C%MOD;
}
ll F(int x,int y){
	int i,last;
	ll re=0;
	for(i=1;i<=x;i=last+1){
		last=min(x/(x/i),y/(y/i));
```

```
            re+=S(x/i)*S(y/i)%MOD*(sum[last]-sum[i-1])%MOD;
                    re%=MOD;
            }
            return re;
}
int main(){
            int i,last;
            cin>>n>>m;
            getmu();
            if(n>m) swap(n,m);
            for(i=1;i<=n;i=last+1){
            last=min(n/(n/i),m/(m/i));
                  ans+=F(n/i,m/i)*(S(last)-S(i-1))%MOD;
                        ans%=MOD;
            }
            cout<<(ans+MOD)%MOD<<endl;
            return 0;
}
```

### 3.4.5 例题 6:

BZOJ 2693: jzptab ~~就是上题改变为多组数据~~ 所以要对上面的式子进行优化

$$\sum_{d=1}^{n} d \sum_{i=1}^{\frac{n}{d}} \mu(i) i^2 \frac{\frac{n}{di}(\frac{n}{di}+1)}{2} \frac{\frac{m}{di}(\frac{m}{di}+1)}{2}$$

$$\begin{aligned}
&\text{设} T = di, \text{则原式变为} \\
&ans = \sum_{T=1}^{n} \frac{\frac{n}{T}(\frac{n}{T}+1)}{2} \frac{\frac{m}{T}(\frac{m}{T}+1)}{2} T \sum_{i|T} \frac{T}{i} \mu(i) i \\
\text{设} f(T) &= \sum_{i|T} \mu(i) i \\
&ans = \sum_{T=1}^{n} \frac{\frac{n}{T}(\frac{n}{T}+1)}{2} \frac{\frac{m}{T}(\frac{m}{T}+1)}{2} T f(T)
\end{aligned}$$

因为 $f(T)$ 同样是个积性函数。
所以 $f(T)$ 进行线性筛，可以 $O(\sqrt{n})$ 处理每一次询问
具体方式如下：线性筛中，外层为 k，内层为 p

当 $p|k$ 时
(1). 当 i 取的数的因子中不包含新加入的 p 时，答案就是 $f(k)$
(2). 当 i 取包含因子 p 时，由于此时 p 指数大于 1，所以 $(i)=0$，无贡献
综上，当 p|k 时，答案为 f(k)

当 $p \nmid k$ 时
(1). 当 i 取的数的因子中不包含新加入的 p 时，答案就是 $f(k)$
(2). 当 i 取的数的因子包含新加入的 p 时, 答案为 $-pf(k)$
证明第二条结论：

$$\begin{aligned}
\text{原式} &= \sum_{i|T} \mu(i) i \\
&= \sum_{ap|kp} \mu(ap) ap = p \sum_{a|k} \mu(a) \mu(p) a \\
&= \sum_{a|k} \mu(a) a = -pf(k)
\end{aligned}$$

综上，当 p∤k 时，答案为 (1–$p$)$f(k)$

```
#include<cstdio>
#include<cstring>
#include<iostream>
#include<algorithm>
#define M 10001000
#define MOD 100000009
using namespace std;
typedef long long ll;

int prime[1001001],tot;
bool not_prime[M];
ll h[M],sum[M];

void Linear_Shaker(){
        int i,j;
        h[1]=1;
        for(i=2;i<M;i++){
                if(!not_prime[i]){
                        prime[++tot]=i;
                        h[i]=(i-(ll)i*i)%MOD;
                }
                for(j=1;prime[j]*i<M;j++){
                        not_prime[prime[j]*i]=1;
                        if(i%prime[j]==0){
                h[prime[j]*i]=(prime[j]*h[i])%MOD;
                                break;
                        }
                h[prime[j]*i]=(h[prime[j]]*h[i])%MOD;
                }
        }
        for(i=1;i<M;i++)
                sum[i]=(sum[i-1]+h[i])%MOD;
}
inline ll Sum(ll x,ll y){
        x%=MOD;y%=MOD;
        ll re1=(x*(x+1)>>1)%MOD;
        ll re2=(y*(y+1)>>1)%MOD;
        return re1*re2%MOD;
}
```

```
int Query(int n,int m){
        int i,last,re=0;
        if(n>m) swap(n,m);
        for(i=1;i<=n;i=last+1){
                last=min(n/(n/i),m/(m/i));
        re+=Sum(n/i,m/i)*(sum[last]-sum[i-1])%MOD;
                re%=MOD;
        }
        return (re+MOD)%MOD;
}
int main(){
        int T,n,m;
        Linear_Shaker();
        for(cin>>T;T;T--){
                scanf("%d%d",&n,&m);
                printf("%d\n",Query(n,m));
        }
}
```