题目

[题目链接]

初步转化

令$f(n)$表示n的所有因数和

如果$d|i$并且$d|j$，那么$d|gcd(i,j)$

那么对于每一个询问，答案为：$\sum_{i=1}^{n}\sum_{j=1}^{m}[f(gcd(i,j))<=a]*f(gcd(i,j))$

所以问题就转化为了回答上述表达式的多组询问

分析

莫比乌斯反演

下面开始用莫比乌斯反演来推式子：

$\sum_{i=1}^{n}\sum_{j=1}^{m}[f(gcd(i,j))<=a]*f(gcd(i,j))$

$=\sum_{d=1}^{n}[f(d)<=a]\sum_{d|i}^{n}\sum_{d|j}^{m}[gcd(i,j)=d]d$

$=\sum_{d=1}^{n}[f(d)<=a]\sum_{i=1}^{n/d}\sum_{j=1}^{m}\sum_{k|gcd(i,j)}\mu(k)d$

$=\sum_{d=1}^{n}[f(d)<=a]d\sum_{k=1}^{n/d}\mu(k)\sum_{k|i}^{n/d}\sum_{k|j}^{m/d}1$

最后两个和式把他变成除法，ij同除k

$=\sum_{d=1}^{n}[f(d)<=a]d\sum_{k=1}^{n/d}\mu(k)*n/kd*m/kd$

令T=kd

$=\sum_{T=1}^{n}n/T*m/T\sum_{d|T}\mu(T/d)*f(d)[f(d)<=a]$

前面一部分可以用整除分块

后面一部分$\sum_{d|T}\mu(T/d)*f(d)[f(d)<=a]$

是f和μ函数的迪利克雷卷积外加一个限制条件

第一部分通过莫比乌斯反演转化式子就完成了

积性函数处理

现在我们难以处理的是a这个限制条件。

设$\sum_{d|T}\mu(T/d)*f(d)[f(d)<=a]$这个函数为G函数

大体思路：那么我们可以考虑离线处理答案，按照询问的a从小到大排序，这样每次a只会变大，那么把没处理的G函数加进去就可以了。

莫比乌斯函数我们可以线性预处理

f函数可以$o(nlogn)$预处理（其实有一种$o(n)$的方法）

a的条件变大了之后，假设x_1是新增进去的，那么就把$\mu(T/x_1)*f(x_1)$加到G(T)里面，其中T是x_1的倍数。

那么现在我们需要一个数据结构来支持：单点修改+前缀查询——树状数组

代码

这样的话思路就理清了：对于询问：按a排序，预处理两个函数，在a变大的同时修改树状数组；树状数组记录G函数。统计答案然后输出就好了

下附AC代码：

```
#include<bits/stdc++.h>
using namespace std;
long long read(){
    char s;
    long long x=0,f=1;
    s=getchar();
    while(s<'0'||s>'9'){
        if(s=='-')f=-1;
        s=getchar();
    }
    while(s>='0'&&s<='9'){
        x*=10;
        x+=s-'0';
        s=getchar();
    }
    return x*f;
}
const long long mod=(long long)1<<31;
const int M=2e4+5;
const int N=1e5+5;
struct Q{
    long long n,m,a;
    int id;
}q[M],di[N];
bool operator<(Q a,Q b){
    return a.a<b.a;
}
bool flag[N];
long long mu[N],d[N];//莫比乌斯函数、约数个数函数
int p[N],pn;
void init(int n){
    mu[1]=1;//mu线性筛  d nlogn筛
    for(int i=2;i<=n;i++){
        if(!flag[i]){
            mu[i]=-1;
            p[pn++]=i;
        }
        for(int j=0;j<pn,p[j]*i<=n;j++){
            flag[p[j]*i]=1;
            if(i%p[j]==0){
                mu[i*p[j]]=0;
                break;
            }
```

```
            else{
                mu[i*p[j]]=-mu[i];
            }
        }
    }
    for(int i=1;i<=n;i++){
        for(int j=i;j<=n;j+=i){
            d[j]+=i;
        }
    }
    for(int i=1;i<=n;i++){
        di[i].id=i;
        di[i].a=d[i];
    }
    sort(di+1,di+1+n);
}
struct tree{
    long long c[N];//记录G函数 即mu和d的卷积函数+d<=a的限制条件
    void clear(){memset(c,0,sizeof(c));}
    int lowbit(int x){return x&(-x);}
    void modify(int pos,long long x){
        for(int i=pos;i<=N-5;i+=lowbit(i)){
            c[i]+=x;
        }
    }
    long long query(int pos){
        long long rec=0;
        for(int i=pos;i>0;i-=lowbit(i)){
            rec+=c[i];
        }
        return rec;
    }
}t;
long long rec[N];
void add(int x){
    for(int i=x;i<=N-5;i+=x){
        t.modify(i,mu[i/x]*d[x]);
    }
}
long long solve(int n,int m){
    long long ans=0;
    if(n>m)swap(n,m);
    for(int l=1,r;l<=n;l=r+1){
        r=min(m/(m/l),n/(n/l));
        ans+=(long long)(t.query(r)-t.query(l-1))*(n/l)*(m/l);
    }
    return ans%mod;
}
int main(){
    int T=read();
    for(int i=1;i<=T;i++){
        q[i].n=read(),q[i].m=read(),q[i].a=read();
        q[i].id=i;
    }
    sort(q+1,q+1+T);
    init(N-5);
    t.clear();
    int id=0;
```

```
    for(int i=1;i<=T;i++){
        while(di[id+1].a<=q[i].a&&id+1<=N-5){
            id+=1;
            add(di[id].id);
        }
        rec[q[i].id]=solve(q[i].n,q[i].m);
    }
    for(int i=1;i<=T;i++){
        long long ans=rec[i]%mod;
        ans+=mod;
        ans%=mod;
        printf("%lld\n",ans);
    }
}
```