线性筛

详细见代码。

用途：O(n)筛积性函数和筛质数。

基本思想：线性筛的线性体现在使每个数只被“筛”一次

关于复杂度：

线性筛的关键就在于这句话if(i%prime[ii]==0)break;

传统的筛法之所以慢，是因为每个数会被筛多次，而线性筛的每一个数只会被它的最小质因子筛一次，因为如果一个数x被非最小质因数筛掉了，设该数x=prime\*w,那么w一定为x的最小质因数的倍数，那么它一定不会有机会\*prime,因为它会在乘上最小质因子后被筛掉，顾线性筛是线性的。例如：12只会被质因数2筛掉，而不会被3筛掉。

筛积性函数的套路：

讨论三种情况：设该函数为f[i]

1.当i为质数时，f[i]的值

2.当i为prime\*ii,其中gcd(prime,ii)==1时,f[i]的值。

3.当i为prime\*ii,其中ii为prime的倍数时f[i]的值。

莫比乌斯函数

μ(i)(以下用u代替μ):

1.u[1]=1.

2.u[w]=(-1)^k,w为k个不同质数的乘积.

3.其他情况均为0.

那么我们如何筛？

考虑上一页的3种情况。

首先当i为质数，u[i]显然为-1.

然后当i为prime\*ii,其中gcd(prime,ii)==1时,u[i]=-u[ii],考虑u第二条性质可得。

最后当i为prime\*ii,其中ii为prime的倍数时,u[i]为0。

那么我们就可以筛了。

void get\_mu()

{

mu[1]=1;

int mn=100000;

for(int i=2;i<=mn;i++)

{

if(!inq[i])

{

prime[++tot]=i;

mu[i]=-1;

}

for(int ii=1;ii<=tot&&prime[ii]\*i<=mn;ii++)

{

inq[prime[ii]\*i]=1;

if(i%prime[ii]==0)

{

mu[i\*prime[ii]]=0;

break;

}

mu[prime[ii]\*i]=-mu[i];

}

}

}

莫比乌斯反演

考虑这样一个函数， ,那么我们如何通过F[n]求出f[n]呢？这就要用到莫比乌斯反演了：

证明：

这里要用到

该式的证明：

当n==1时显然，

当n!=1时，首先分解n为,首先u不为0的仅有ai=1或ai=0时的情况，那么原式=

考虑二项式定理：

我们令x=-1,y=1即可证明。

下面我们找一道例题：

对于给出的t个询问，每次求有多少个数对(x,y)，满足a≤x≤b，c≤y≤d，且gcd(x,y) = k，gcd(x,y)函数为x和y的最大公约数。

100%的数据满足：1≤t≤50000，1≤a≤b≤50000，1≤c≤d≤50000，1≤k≤50000

暴力大家都会吧？直接枚举即可。

首先利用容斥原理将一个询问拆分成四个，每次求有多少个数对(x,y)，满足1≤x≤n，1≤y≤m，且gcd(x,y) = k。

f(i)为1<=x<=n,1<=y<=m且gcd(x,y)=i的数对(x,y)的个数，F(i)为1<=x<=n,1<=y<=m且i|gcd(x,y)的数对(x,y)的个数，那么

那么

易得

这样我们就可以O(n)完成了，但还是不够快

观察到只有sqrt(n)种取值，那么我们需要对莫比乌斯函数存一个前缀和即可，即可以在即可出答案。那么我们如何快速枚举这几种取值呢？

ll getans(int n,int m)

{

int last=0;

ll ans=0;

for(int i=1;i<=n;i=last+1)

{

last=min(n/(n/i),m/(m/i));

ans=(ans+(mu[last]-mu[i-1])\*(n/last)\*(m/last));

}

return ans;

}