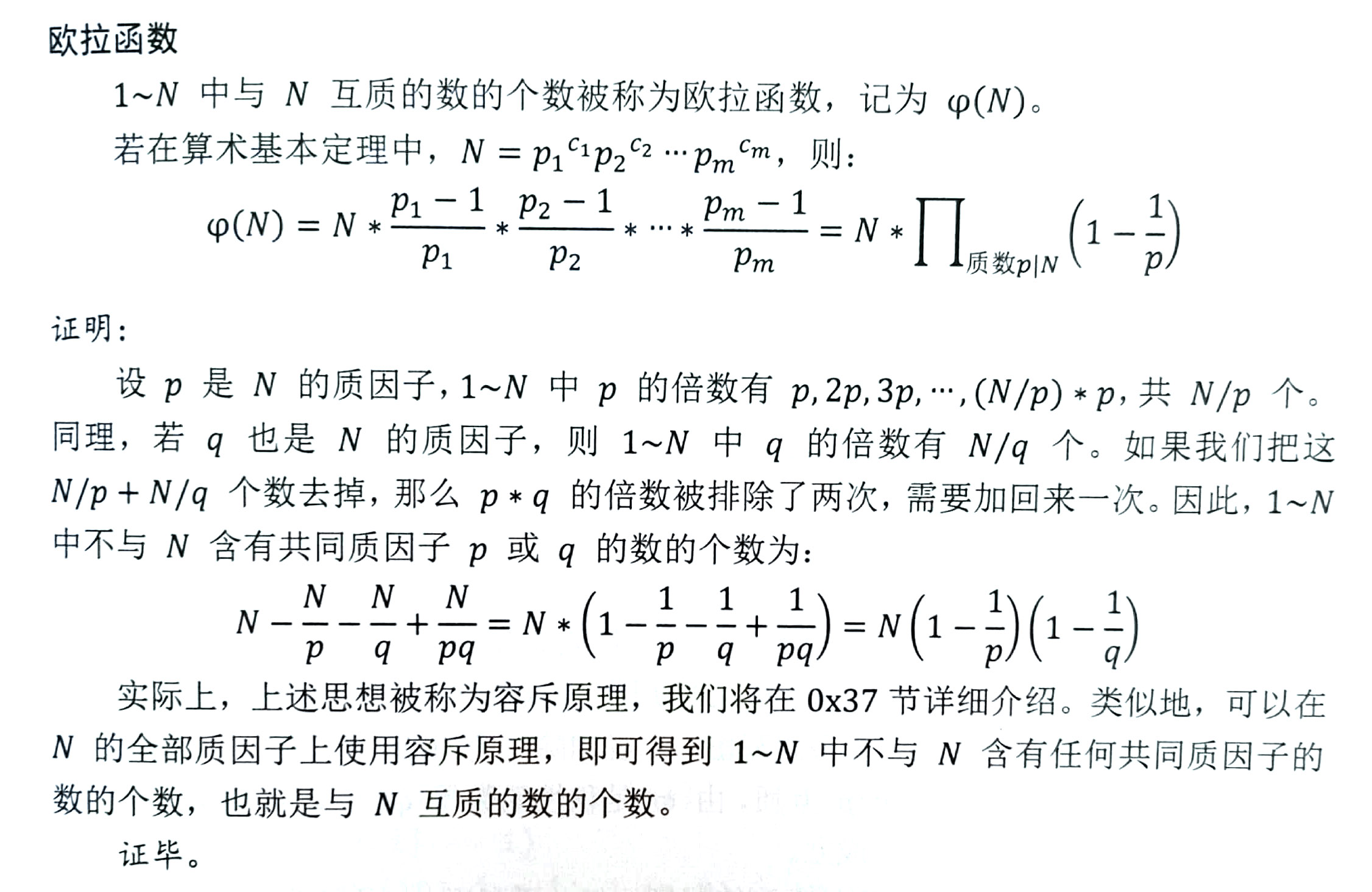
# 欧拉函数



1.含义：对φ(N)的值,我们可以通俗地理解为小于N且与N互质的数的个数(包含1).

2.性质：a.对于一个正整数N的素数幂分解N=P1^q1\*P2^q2\*...\*Pn^qn.

              φ(N)=N\*(1-1/P1)\*(1-1/P2)\*...\*(1-1/Pn).

     b.φ(1)=1，除了N=2,φ(N)都是偶数.

     c.小于n与n互质的所有数的和为 φ（n）\*n/2;

     d.p为素数，φ（p^k）=p^k-p^(k-1).

             H（p）=p-1;

     e. 如果m和n互素，即GCD(m, n) = 1，那么φ(m \* n) = φ(m) \* φ(n);

     f.欧拉降幂公式：IMG_256

3.求解欧拉函数值：

第一种：φ(N)=N\*(1-1/P1)\*(1-1/P2)\*...\*(1-1/Pn).  O(sqrt(n))

int euler(int n){ *//返回euler(n)*

int res=n,a=n;

for(int i=2;i\*i<=a;i++){

if(a%i==0){

res=res/i\*(i-1);*//先进行除法是为了防止中间数据的溢出*

while(a%i==0) a/=i;

}

}

if(a>1) res=res/a\*(a-1);

return res;

}

第二种：线性筛选法

φ(n)=n\*（1-1/p1)(1-1/p2)....(1-1/pk),其中p1、p2…pk为n的所有素因子。  
比如：φ(12)=12\*(1-1/2)(1-1/3)=4。  
利用这个就比较好求了，可以用类似求素数的筛法。  
先筛出N以内的所有素数，再以素数筛每个数的φ值。  
比如求10以内所有数的φ值：  
设一数组phi[11]，赋初值phi[1]=1,phi[2]=2...phi[10]=10；  
然后从2开始循环，把2的倍数的φ值\*(1-1/2)，则phi[2]=2\*1/2=1,phi[4]=4\*1/2=2,phi[6]=6\*1/2=3....；  
再是3，3的倍数的φ值\*(1-1/3)，则phi[3]=3\*2/3=2,phi[6]=3\*2/3=2，phi[9]=.....；  
再5，再7...因为对每个素数都进行如此操作，因此任何一个n都得到了φ(n)=n\*（1-1/p1)(1-1/p2)....(1-1/pk)的运算  
觉得这个“筛”还是比较好用的，以前求数的所有因子之和也是用的它。  
代码如下：

int pr[maxn/5],p[maxn+100],tot,phi[maxn+100];

void init(){

phi[1]=1;p[1]=1;

for(int i=2;i<2\*maxn;i++) {

if (!p[i]) p[i]=i,pr[++tot]=i,phi[i]=p[i]-1;

for (int j=1;j<=tot&&pr[j]\*i<=2\*maxn;j++) {

p[i\*pr[j]]=pr[j];

if (p[i]==pr[j]){

phi[i\*pr[j]]=phi[i]\*pr[j];

break;

} else phi[i\*pr[j]]=phi[i]\*(pr[j]-1);

}

}

}

4.练习

## **HODJ 3501**

小于N并且不互质的数字和

注意：判断互质不能用N%i==0，比如6，4,判断互质的方式是GCD

N的范围是32位整数范围

直观想法： 所有小于n且与n为非互质数和=所有小于n数的和-所有小于n且与n互质的数的和

原理：  
 求所有小于N且与N为互质数的和：

        1.欧拉函数可求与N互质的数的个数  
       ****2.if gcd(n,i)=1 then gcd(n,n-i)=1 (1<=i<=n)。若已知m与n互质，则n-m也与n互质****

那么，对于任何一个i与n互质，必然n-i也和n互质，所以PHI（N）必然是偶数（除了2）

所以从1到N与N互质的数的和为PHI(N)\*N/2（一对一对算）

## ****HDOJ2588****

来源：http://blog.csdn.net/ydd97/article/details/47858679

给定N,M求gcd(i,N)>=M的i的个数

我们可以分解N=a\*b, i=a\*d(b>=d 且b，d互质)，那么我们要求的就是a》=m的时候d的个数（b随a而确定）

由于b>=d且b，d互质，所以这个数目就是φ（b）-1

但是，如果对于每个a枚举b，铁定超时。（仍然O((N-M)\*sqrt(N))的复杂度）

但是如果单纯这样全部枚举的话依旧会超时，所以我们要想一个办法去优化它。

我们可以折半枚举，这里的折半并不是二分的意思。

我们先看，我们枚举时，当i<sqrt(n),假设a=n / i， 当i>sqrt(n)之后 有b=n/i，我们观察到当n%i==0时，会出现一种情况，就是a\*b==n。所以我们就可以只需要枚举sqrt(n)种情况，然后和它对应的情况就是 n/i。

<https://blog.csdn.net/yu121380/article/details/81296803>

<https://www.cnblogs.com/wuwangchuxin0924/p/6233211.html>