

# 数论模板：

## 质因数：

### 1.筛质数：

- 埃筛 $O(n\log)$ :

```
#include <iostream>
#include <algorithm>

using namespace std;
//埃氏筛 通过n之前所有质数标记合数
// 时间复杂度 调和级数  $n\log(\log n)$ 
const int N= 1000010;

int prime[N], cnt=0;
bool vis[N]; // 标记合数
void get_primes(int n)
{
    for(int i=2;i<=n;i++){
        if(!vis[i]){ //未被标记 prime
            prime[++cnt]=i;
            for(int j=i+i;j<=n;j+=i) vis[j]=1; //通过 质数j标记之后合数
        } //这里可从i*i开始
    }
}

int main()
{
    int n;
    cin >> n;
    get_primes(n);
    cout << cnt << endl;
    //for(int i=1;i<=cnt;i++)cout<<prime[i]<<" ";

    return 0;
}
```

- 线性筛\*\*\*接近 $O(n)$

```
#include <bits/stdc++.h>
//#define LOCAL
//#define int long long
//线性筛--欧拉筛 更快
//方法 用n的最小质因子筛n*** (防重复)
const int N=1e6+5;
int prime[N], vis[N], cnt;

void get_prime(int n){
    for(int i=2;i<=n;i++){
        if(!vis[i]) prime[++cnt]=i; //未标记就加入
```

```

        for(int j=1;prime[j]<=n/i;j++){// **每次用 prime[j] 作k=prime[j]*i 的最小质
因子筛k
            vis[i*prime[j]]=1;
            if(i%prime[j]==0)break; //prime[j]已是i的最小质因子 不能继续 优化为线性
(防止重复标记)
        }
    }
}
using namespace std;
int T,n;
signed main(){
    cin>>n;
    get_prime(n);
    cout<<cnt<<endl;
    //for(int i=1;i<=cnt;i++)cout<<prime[i]<<" ";
    return 0;
}

```

## 2.求欧拉函数

### 欧拉函数：

$\phi[i]$ 表示1~i中与i互质的数的个数

公式：  $\phi(n) = n * (1 - 1/p_1) * (1 - 1/p_2) * \dots * (1 - 1/p_n)$

### 欧拉函数性质：

- 二.欧拉函数的一些性质
  - 若n为质数，则  $\phi(n) = n - 1$ ;
  - 若m与n互质，则  $\phi(n*m) = \phi(n) * \phi(m)$ ;
  - 若正整数n与a互质，那么就有

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

若n为奇数时，  $\phi(2n) = \phi(n)$ ;

若  $n = p^k$  且p是质数，那么  $\phi(n) = (p - 1) * p^{(k-1)} = p^k - p^{(k-1)}$ .

推导：容斥原理：i先减去i内所有质数个数，再加上(上一步多减了)两质数积的倍数，再减去.....

- 公式求欧拉函数

```

#include<bits/stdc++.h>
using namespace std;
//运用公式即可
int n;
int main(){
    cin>>n;
    int ans=n;
    for(int i=2;i<=n/i;i++){
        if(a%i==0){
            ans=ans/i*(i-1); //公式 注意写法 防止不整除
            while(n%i==0)n/=i;
        }
    }
}

```

```

    }
}
// 别漏了
if(n>1)ans=ans/n*(n-1);
cout<<ans<<endl;
return 0;

```

- 线性筛O(n)求1~n欧拉函数值

```

#include<bits/stdc++.h>
using namespace std;
#define LL long long
const int N=1e6+5;
//求1~n 欧拉函数之和
int n;
int prime[N],vis[N];
int phi[N];
int main(){
    cin>>n;
    LL res=0;
    int cnt=0;
    phi[1]=1; // 特殊
    for(int i=2;i<=n;i++){
        if(!vis[i]){
            prime[++cnt]=i;
            phi[i]=i-1; //质数
        }
        for(int j=1;prime[j]<=n/i;j++){
            vis[prime[j]*i]=1; // 线性筛
            if(i%prime[j]==0){
                phi[prime[j]*i]=prime[j]*phi[i]; //线性筛同时求欧拉函数
                break; //线性筛优化
            }
            else phi[prime[j]*i]=(prime[j]-1)*phi[i];
        }
    }
    for(int i=1;i<=n;i++){
        res+=phi[i];
    }
    cout<<res<<endl;
    return 0;
}

```

## 逆元、线性同余方程组：

### 逆元：

对于正整数a和p，如果有  $ax \equiv 1 \pmod{p}$ ，那么称x的最小整数解为a模p的逆元

### 欧拉定理

若正整数n与a互质，那么就有

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

特殊:

当 $p, a$ 互质且 $p$ 为质数, 有:

## 费马小定理: $a^{(p-1)} \equiv 1 \pmod{p}$

即  $a \cdot a^{(p-2)} \equiv 1$  由逆元定义: 得到

结论: 若 $a$ 与 $p$ 互质,  $p$ 为质数,  $a^{(p-2)}$ 是 $a$ 逆元

## 线性同余方程

## 扩展欧几里得算法

```
#include <bits/stdc++.h>
// #define LOCAL
// #define int long long
// 扩展欧几里得算法 求 ax同余b 模m --> ax+my=b=d*k
// b 为gcd 倍数 有解 反之无解 逆元 (d==1)
int m, a, b;
int ex_gcd(int a, int b, int &x, int &y){
    if(b==0){ // 跳出条件
        x=1;
        y=0;
        return a; //
    }
    int gcd=ex_gcd(b, a%b, y, x);
    y-=a/b*x;
    return gcd; //
}
using namespace std;
int T, n;
signed main(){
    cin>>T;
    while(T--){
        cin>>a>>b>>m;
        int x, y; // 不用赋值
        int d=ex_gcd(a, m, x, y); //
        if(b%d==0)
            cout<<x*(long long)b/d%m<<endl; // 放大
        else cout<<"impossible"<<endl;
    }
    return 0;
}
```