Chapter 1

数论

1.1 指数降幂公式

$$A^x \equiv A^{x \bmod \phi(p)+\phi(p)} \pmod p (x \geq \phi(p))$$

1.2 威尔逊定理

$$(p-1)! \equiv -1 \pmod p$$

1.3 费马小定理

$$a^p \equiv a \pmod p$$

1.4 欧拉定理

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

1.5 质数表

```
const int N = 1000000 + 9;
bool p[N];
int a[N];
int main()
{
  int n;
  cin >> n;
  int cnt = 0;
  for (int i = 0; i <= n; i++) p[i] = true;
  for (int i = 2; i <= n; i++) {
```

```
    if (p[i]) a[cnt++] = i;
    for (int j = 0; j < cnt; j++) {
        if (i * a[j] > n) break;
        p[i * a[j]] = false;
        if (i % a[j] == 0) break;
    }
  }
  cout << cnt << endl;
}
```

1.6 素数函数

```
#include <bits/stdc++.h>
typedef long long ll;
using namespace std;
ll f[340000],g[340000], n;
void init(){
    ll i, j, m;
    for(m = 1; m * m <= n; ++m)f[m]=n/m-1;
    for(i=1;i<=m;++i)g[i]=i-1;
    for(i=2;i<=m;++i){
        if(g[i]==g[i-1])continue;
        for(j=1;j<=min(m-1,n/i/i);++j){
            if(i*j<m)f[j]-=f[i*j]-g[i-1];
            else f[j]-=g[n/i/j]-g[i-1];
        }
        for(j=m;j>=i*i;--j)g[j]-=g[j/i]-g[i-1];
    }
}
int main()
{
    while (cin >> n) {
      init();
      cout << f[1] << endl;
    }
}
```

1.7 欧拉函数

1.7.1 递推求

```
// 傻逼写的
int phi[MAXN];
```

```
void init() {
    memset(phi, 0, sizeof(phi));
    phi[1] = 1;
    for(int i = 2; i < MAXN; i++) if(!phi[i])
        for(int j = i; j < MAXN; j += i) {
            if(!phi[j]) phi[j] = j;
            phi[j] = phi[j] / i * (i - 1);
        }
}
// 聪明人写的
void init()
{
  for (int i = 2; i < maxb; i++)
    if (f[i] == 0)
      for (int j = i; j < maxb; j += i)
      {
        if (f[j] == 0) f[j] = j;
        f[j] = (f[j] / i) * (i - 1);
      }
  for (int i = 1; i < maxb; i++)
    f[i] += f[i - 1];
}
```

1.7.2 单个求

```
ll phi(ll n){
    ll ans = n,a = n;
    for(ll i = 2; i * i <= a; i++){
        if(a % i == 0){
            ans = ans / i * (i - 1);
            while(a % i == 0) a /= i;
        }
    }
    if(a > 1) ans = ans / a * (a - 1);
    return ans;
}
```

1.8 莫比乌斯函数

```
int mu[MAXN], prm[MAXN], vis[MAXN];
void getmu(int n) {
    int sz = 0;
    mu[1] = 1;
```

```
    for(int i = 2; i <= n; i++) {
        if(!vis[i]) prm[++sz] = i, mu[i] = -1;
        for(int j = 1; j <= sz && prm[j] * i <= n; j++) {
            vis[i * prm[j]] = 1;
            if(i % prm[j] == 0) {mu[i * prm[j]] = 0; break;}
            mu[i * prm[j]] = -mu[i];
        }
    }
}
```

1.9 逆元

1.9.1 递推求

```
ll mul_mod(ll x, ll y, ll mod = MOD)
{
    return x * y % MOD;
}
m[1] = 1;
inv[1] = 1;
for (n = 2; ;n++) {
    s[n] = s[n - 1] + n;
    m[n] = mul_mod(m[n - 1], n);
    inv[n] = (MOD - MOD / n) * inv[MOD % n] % MOD;
    if (s[n] > MAXX) break;
}
```

1.9.2 单个求

用费马小定理