数论

筛质数

朴素筛法

从 2 开始遍历到 n ，将每个数的倍数删去，留下了的数就是质数

解释：

设 p 为质数，说明在枚举 2 到 $p-1$ 的过程中 p 都没有被删去，即 2 到 $p-1$ 中没有一个数是 p 的因数，因此 p 是质数

完整代码：

```
const int N = 1e6 + 10;
bool st[N];//表示当前数是否为质数
int cnt;//质数的数量
int primes[N];//存储质数的集合

void get_primes(int n)//所有1到n中的质数
{
    for(int i = 2; i <= n; i ++)
    {
        if(!st[i])
            primes[cnt++] = i;
        for(int j = i + i; j <= n; j += i)
            st[j] = true;
    }
}
```

埃氏筛法

由算术基本定理可知，任何一个大于 1 的自然数 n ，如果不是质数，那么必然可以被质因数分解

因此在晒质数的过程中，只需要删去**每个质数的倍数**即可，也就是把第二重循环移到`if`内部

完整代码：

```
void get_primes(int n)//所有1到n中的质数
{
    for(int i = 2; i <= n; i ++)
    {
        if(!st[i])
        {
            primes[cnt++] = i;
            for(int j = i + i; j <= n; j += i)
                st[j] = true;
```

```
        }
    }
}
```

线性筛

对于每个合数，如果它**只会被最小的质因子筛掉**，那么它只会被筛掉一次

也就是对于当前枚举的数 i ，枚举所有质数 $primes[j]$

- 如果 i 不能整除 $primes[j]$ 说明 i 与 $primes[j]$ 都是 $i \times primes[j]$ 的**最小质因子**
- 如果 i 能够整除 $primes[j]$ 说明存在更小的 i' 使得当前的 $i \times primes[j]$ 能够被**提前**删去，也就是当出现这种清空，我们需要跳出循环

完整代码：

```
void get_primes(int n)
{
    for(int i = 2; i <= n; i ++)
    {
        if(!st[i]) primes[cnt++] = i;
        for(int j = 0; primes[j] <= n / i; j ++)
        {
            st[i * primes[j]] = true;
            if(i % primes[j] == 0) break;
        }
    }
}
```