质因数分解

方案一

枚举从 2 到 a 的每个数，如果 i 是 a 的一个因数，尝试把他除去并累计幂次。

```
for (int i = 2; i <= a; i++) {
    while (a % i == 0) {
        a /= i;
        mp[i]++;
    }
}
```

方案二

在方案一的基础上进行小优化，注意到：

> 引理
>
> 如果 a 有大于 $\sqrt{a}$ 的质因数，那么它一定是唯一的。
>
> > 证明
> >
> > 反证法，如果有两个大于 $\sqrt{a}$ 的质因数，相乘后**一定大于** a。

```
for(int i=2; i*i<=a; i++) {
    while(a%i == 0) {
        a /= i;
        mp[i]++;
    }
}
if(a > 1) mp[a]++;  // 处理最后剩下的质因数
```

方案三

先打质数表再比较，比较复杂但是快一些。

以下段落为 AI 辅助生成，不保证正确性。

```
const int MAXN = 50000000;
bool is_prime[MAXN + 1];
vector<int> primes;
void sieve() {
    fill(is_prime, is_prime + MAXN + 1, true);
    is_prime[0] = is_prime[1] = false;
    for (int i = 2; i * i <= MAXN; ++i) {
        if (is_prime[i]) {
            for (int j = i * i; j <= MAXN; j += i) {
                is_prime[j] = false;
            }
        }
    }
    for (int i = 2; i <= MAXN; ++i) {
        if (is_prime[i]) {
            primes.push_back(i);
        }
    }
}
```

然后用质数表进行分解：

以下段落为 AI 辅助生成，不保证正确性。

```
for (int prime : primes) {
    if (prime * prime > a) break;
    while (a % prime == 0) {
        a /= prime;
        mp[prime]++;
    }
}
if (a > 1) mp[a]++;
```

`sieve()` 生成质数表后，可多次用于分解多个数，适用于需要频繁分解的场景。

Application 应用

求因子和

公式的展开与推导思路

设 $n = p^a \cdot q^b \cdot r^c \cdots$，其中 $p, q, r, \ldots$ 是不同的质数。

1. 单个质数幂的情况：

对于 p^a，它的所有因数是：$1, p, p^2, \ldots, p^a$。

这是一个等比数列，其和为：

$$\sigma(p^a) = 1 + p + p^2 + \cdots + p^a = \frac{p^{a+1} - 1}{p - 1}$$

这就是公式中每个因子的来源。

2. 多个质数幂相乘的情况：

当 $n = p^a \cdot q^b$ 时，n 的每一个因数都可以**唯一地**写成 $p^x \cdot q^y$ 的形式，其中 $0 \le x \le a$，$0 \le y \le b$。

因此，所有因数的和就等于：

$$\sigma(n) = (1 + p + p^2 + \cdots + p^a) \times (1 + q + q^2 + \cdots + q^b)$$

因为第一个括号里的每一项（p的幂）都要与第二个括号里的每一项（q的幂）相乘，从而生成所有可能的因数。

3. 推广到一般形式：

对于一般的质因数分解 $n = p_1^{a_1} \cdot p_2^{a_2} \cdots p_k^{a_k}$，其所有正因数的和就是每个质因数幂的（等比数列）因数和之**乘积**：

$$\boxed{\sigma(n) = \prod_{i=1}^{k} \left(1 + p_i + p_i^2 + \cdots + p_i^{a_i}\right) = \prod_{i=1}^{k} \frac{p_i^{a_i+1} - 1}{p_i - 1}}$$