数论专题

质数

河南省实验中学信息技术组

2026 年 02 月 03 日

数论专题

- 质数
- 约数
- 同余

质数定义

- 若一个大于等于 1 的正整数无法被除了 1 和它自身之外的任何自然数整除，则称该数为质数，否则称该正整数为合数。
- 整个自然数集合中，质数数量不多，分布比较稀疏，对于一个足够大的整 n，不超 n 的质数大约有 $\frac{n}{\ln n}$ 个，即每 $\ln n$ 个数中大约有一个质数。

质数判定

- 试除法 1
  - 根据定义判断 $2 \sim n-1$ 是否存在能整除 n 的数。
  - 时间复杂度：$\mathrm{O}(N)$
- 试除法 2
  - 判断 $2 \sim \sqrt{n}$ 中是否存在能整除 n 的数。
  - 自然数的约数都是成对出现的，如果在 $2 \sim \sqrt{n}$ 中没有 n 的约数，那么在 $\sqrt{n}+1 \sim n-1$ 中也不会有 n 的约数。
  - 时间复杂度 O：$(\sqrt{N})$

```cpp
bool prime(int n)
{
    if(n == 1) return false;
    for(int i = 2; i * i <= n; ++i)
        if(n % i == 0) return false;
    return true;
}
```

质数筛选

- 给定一个整数 n，求出 $1 \sim n$ 之间所有的质数。
- 算法 1：直接求解
  - 枚举 $2 \sim n$ 之间所有的数，判断其是否为质数。
  - 时间复杂度：$O(N\sqrt{N})$

埃氏筛法

- 埃氏筛 (Eratosthenes 筛法) 的思想：任意正整数 x 的倍数 $2x, 3x, \cdots$ 都不是质数。
- 从 2 开始遍历每一个数 x，如果它是质数，把它的倍数 $2x, 3x, \cdots, \lfloor \frac{N}{x} \rfloor x$ 标记为合数。
- 当扫描到一个数 x 时，若它尚未标记，则它不能被 $2 \sim x-1$ 之间的任何数整除，该数就是质数。
- 埃氏筛的时间复杂度为 $\mathrm{O}(\sum\limits_{p \le N} \frac{N}{p}) = \mathrm{O}(N \log\log N)$，其中 p 是质数。

埃氏筛法

整数x	待筛数据														
2	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
4	4是合数，曾经2被筛过，那么4的倍数一定被2筛过														
5	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
6	6是合数，曾经2被筛过，那么6的倍数一定被2筛过														
7	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
结果	0	0	1	0	1	0	1	1	1	0	1	0	1	1	1

图: 埃氏筛案例

埃氏筛法实现

```
// 合数标记 合数->1 质数->0
memset(v, 0, sizeof(v));
for(int i = 2; i <= n; ++i)
{
    // 如果 i 是合数 那么它的倍数都被筛过 直接跳过
    if(v[i]) continue;
    // 把 i 的 j 数倍标记为合数
    for(int j = 2; j <= n / i; ++j) v[i * j] = 1;
}
```

埃氏筛法改进[1]

- 埃氏筛法改进：对于整数 x，小于 x^2 的 x 的倍数在扫描更小的数时就已经被标记过了。例如：一个数 $x(x-k)$ 小于 x^2 且是 x 的倍数，$x-k$ 是一个质数，那么它一定会被 $x-k$ 筛去。
- 从 2 开始遍历每一个数 x，如果它是质数，把它的倍数 $x^2, x(x+1), \cdots, \lfloor \frac{N}{x} \rfloor x$ 标记为合数。
- 当扫描到一个数 x 时，若它尚未标记，则它不能被 $2 \sim x-1$ 之间的任何数整除，该数就是质数。

整数x	被x筛数据										
11	22	33	44	55	66	77	88	99	110	121	132
7						77					
5				55					110		
3		33			66			99			
2	22		44		66		88		110		

图: 小于 11^2 的数被筛情况

[1] 实际意义不大，自行了解。

埃氏筛法改进实现

质数

河南省实验中学信息技术组


```cpp
// 合数标记 合数->1 质数->0
memset(v, 0, sizeof(v));
for(int i = 2; i <= n; ++i)
{
    // 如果 i 是合数 那么它的倍数都被筛过 直接跳过
    if(v[i]) continue;
    // 把 i 的 j 数倍标记为合数 从 i 倍开始
    for(int j = i; j <= n / i; ++j) v[i * j] = 1;
}
```

质数

河南省实验中学信息技术组

欧式筛引入

整数x	待筛数据																			
2	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
3	从3×3开始筛							9	10	11	12	13	14	15	16	17	18	19	20	21
5	从5×5开始筛，没有筛除新的数																			
被筛次数			1		1	1	1	1	1	1	2	0	1	1	1		2		1	3

整数x	待筛数据																			
2	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
3	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
5				25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
被筛次数	1		2	1	1	1	1		3		1	1	1	1	2		1	1	2	

欧式筛

- 经过优化后的埃氏筛仍会重复筛除合数。例如，12 既会被 2 筛也会被 3 筛，这就出现了重复筛选；再例如，30 被 2、3、5 各筛一次。
- 线性筛法让每一个合数只被其最小质因子筛一次。
- 设数组 v 记录每个数的最小质因子，即该数被哪个数筛，我们按照如下步骤维护 v:
  - 依次考虑 $2 \sim n$ 之间的每一个数 i;
  - 如果 $v[i] = 0$，说明 i 未被筛过，即为质数，把它存下来，并令 $v[i] = i$;
  - 扫描不大于 $v[i]$ 的每个质数 p，令 $v[i * p] = p$。因为 $p \leq v[i]$，所以 p 就是合数 $i * p$ 的最小质因子。
- 因为每个合数 $i * p$ 只会被它的最小质因子 p 筛一次，所以时间复杂度为 $\mathrm{O}(N)$。

欧式筛

i	$v[i]$	$p \le v[i]$	待筛数据，用p来筛除$p*i$																			
2	2	2	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
3	3	2,3	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
4	2	2	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
5	5	2,3,5	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
6	2	2	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
7	7	2,3,5,7	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
8	2	2	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
9	3	2,3	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
10	2	2	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
结果			1	1	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0

欧式筛实现

```cpp
int p[1000010];
int m = 0;  // 质数个数

memset(v, 0, sizeof(v));
for(int i = 2; i <= n; ++i)
{
    if(!v[i]) v[i] = i, p[++m] = i; // i 是质数
    for(int j = 1; j <= m; ++j) // 给当前的数 i 乘上一个质因子 p[j]
    {
        // i 的最小质因子比 p[j] 小或者待筛数超出 n 的范围
        if(p[j] > v[i] || i * p[j] > n) break;
        v[i * p[j]] = p[j]; // p[j] 是合数 i*p[j] 的最小质因子
    }
}

```

算术基本定理

- 任何一个大于 1 的正整数都能唯一分解为有限个质数的乘积，即：

$$n = p_1^{c_1} p_2^{c_2} \cdots p_m^{c_m}$$

其中 c_i 都是正整数，p_i 都是质数，且满足 $p_1 < p_2 < \cdots < p_m$。

- $8 = 2 \times 2 \times 2 = 2^3$
- $30 = 2 \times 3 \times 5 = 2^1 3^1 5^1$
- $36 = 2 \times 2 \times 3 \times 3 = 2^2 3^3$
- $100 = 2 \times 2 \times 5 \times 5 = 2^2 5^2$

质因数分解

- 质因数分解：将一个数按照算术基本定理的形式进行分解。
- 结合质数判定的试除法和埃氏筛法，我们可以扫描 $2 \sim \sqrt{n}$ 的每个数 d，若 d 能整除 n，则从 n 中除掉所有的因子 d，同时累计除去 d 的个数。
- 一个合数的因子一定在扫描到这个合数之前就从 n 中被除掉了，所以在上述过程中能整除 n 的一定是质数。例如，$N = 36$ 时，处理时 d 不可能等于 4，因为在前面 $d = 2$ 时，4 已经被拆分成 2^2 。
- 时间复杂度：$O(\sqrt{N})$

质因数分解实现

- 引理：小于等于 2×10^9 的数的不同质因子个数不会超过 *10* 个，且所有质因子的指数总和不超过 *30*。
- **证明：最小的 11 个质数的乘积** $2\times3\times5\times7\times11\times13\times17\times19\times23\times29\times31>2\times10^9$**，所以对于小于等于** 2×10^9 **的数不可能有多于 10 个质因子。那么，即使包含最小的质数，仍然有** $2^{31}>2\times10^9$**，所以对于小于等于** 2×10^9 **的数的指数总和不可能超过 30。**

```
int p[15], c[15];
int m = 0;
for(int i = 2; i * i <= n; ++i)
{
    if(n % i == 0) // i 是质数
    {
        p[++m] = i, c[m] = 0;
        // 除掉 n 中所有的 i
        while(n % i == 0) n /= i, ++c[m];
    }
}
// n 是质数或者最后除完仅剩一个质数
if(n > 1) p[++m] = n, c[m] = 1;
```

质因数分解

- 对前 n 个数进行质因数分解。
- 方法 1：遍历每一个数，进行试除法，时间复杂度 $\mathrm{O}(N\sqrt{N})$。

质因数分解

- 对前 n 个数进行质因数分解。
- 方法 2：前 n 个数进行质因数分解要求将每个数的质因数和指数都求出来，而在埃氏筛法中，每个数都会被它的所有质因子筛一次。
- 所以，对前 n 个数进行埃氏筛法，对于每一个质数 p，在用 p 筛它的倍数时将这些数的的对应质因子 p 的指数都算出来。
- 时间复杂度：$\mathrm{O}(N \log N)$。

质因数分解

- 对前 n 个数进行质因数分解。
- 在对 n 进行质因数分解时，每次先得到的是其最小质因子的次方。例如，$n = 36$ 时，我们先得到的是 2^2，此时 $n = 9$，最后分解 9 得到 3^2，二者合并起来就得到 $n = 36 = 2^2 3^2$。
- 方法 3：先对前 n 个数进行欧式筛记录每个数的最小质因子；然后对于 $2 \sim n$ 的整数 m，除尽所有的最小质因子，得到新数 $m' < m$，而 m' 的质因数分解在前面的求解过程中已知。例如：$divide(36) = 2^2 divide(9)$。
- 时间复杂度：$\mathrm{O}(N \log N)$。

【例】质数距离

【题目描述】

给定两个整数 $L, R(1 \le L \le R \le 2^{31}, R - L \le 10^6)$ 求闭区间 $[L, R]$ 相邻两个质数的差的最大是多少，输出这两个质数 (输入数据有多组)。

【输入格式】

多行，每行两个整数 L, R。

【输出格式】

多行，每行连个整数表示差最大的两个质数 (小数在前，大数在后)，如果有多组解输出较小的，如果没有解输出 “no adjacent primes”。

【样例输入】

```
2 17
14 17
```

【样例输出】

```
7 11
no adjacent primes
```

【样例解释】

2 到 17 中的质数有 $2, 3, 5, 7, 11, 13, 17$，相差最大的质数对有 $7, 11$ 和 $13, 17$。14 到 17 中的质数只有 17，不存在质数对。

【例】质数距离

- 方法 1：计算出闭区间 $[L,R]$ 中的所有质数，然后求解相邻质数差最大的质数对，时间复杂度 $O((R-L)\sqrt{N})$。

【例】质数距离

- 方法 2：因为 $R-L \le 10^6$ 范围很小，$[L,R]$ 中每一个合数都包括一个 $\sqrt{R}$ 以内的质因子。
  - 利用欧式筛求出 $2 \sim \sqrt{R}$ 之间的所有质数；
  - 对于 $2 \sim \sqrt{R}$ 之间的每一个质数 p，将其在 $[L,R]$ 内的倍数全部筛掉;
  - 所有未标记的都是质数，找出相邻质数差最大的即可。
  - 时间复杂度：$\mathrm{O}(\sqrt{R}+(R-L))$

【例】阶乘分解

【题目描述】

给定正整数 $n(1 \le n \le 10^6)$，试把阶乘 $n!$ 分解质因数，按照算术基本定理的形式输出分解结果中的 p_i 和 c_i 即可。

【输入格式】

一个正整数 n。

【输出格式】

多行，每行两个正整数表示 p_i 和 c_i，要求按照 p_i 从小到大输出。

【样例输入】

```
8
```

【样例输出】

```
2 7
3 2
5 1
7 1
```

【样例解释】

$$8! = 1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 7 \times 8 = 2^1 \times 3^1 \times 2^2 \times 5^1 \times 2^1 \times 3^1 \times 7^1 \times 2^3 = 2^7 3^2 5^1 7^1$$

【例】阶乘分解

- 方法 1：对 $1 \sim n$ 中的每个数分解质因数，将结果合并。
- 时间复杂度：$\mathrm{O}(n\sqrt{n})$。

【例】阶乘分解

- 方法 2：在埃氏筛中，每一个数都会被它的所有质因子筛一次。
- 所以对 $1 \sim n$ 进行埃氏筛法，在筛选过程中，如果数字为质数，那么记录下来，接着用它来筛选所有以它为质因子的合数，此时将合数里所有该质因子求出。
- 时间复杂度：$O(n \log n)$。

质数x	待筛数据							筛出合数时将它对应的质因子求出
2	2	3	4	5	6	7	8	$2^{1+2+1+3} = 2^7$
3	2	3	4	5	6	7	8	$3^{1+1} = 3^2$
5	2	3	4	5	6	7	8	5^1
7	2	3	4	5	6	7	8	7^1

【例】阶乘分解

- 方法 3：$N!$ 的每个质因子不会超过 N，我们可以先筛出 $1 \sim N$ 中的质数，对于质数 p，考虑阶乘中有多少个质因子 p。
  - 在 $1 \sim N$ 中，至少包含 1 个质因子 p 的显然有 $\left\lfloor \frac{N}{p} \right\rfloor$ 个；
  - 在 $1 \sim N$ 中，至少包含 2 个质因子 p^2 的显然有 $\left\lfloor \frac{N}{p^2} \right\rfloor$ 个，此时统计质因子个数不是 $2\left\lfloor \frac{N}{p^2} \right\rfloor$ 个，因为 1 个因子 p 的已经统计过；
  - 所以，$N!$ 中质因子 p 的个数为：

$$\left\lfloor \frac{N}{p} \right\rfloor + \left\lfloor \frac{N}{p^2} \right\rfloor + \left\lfloor \frac{N}{p^3} \right\rfloor + \cdots + \left\lfloor \frac{N}{p^{\left\lfloor \log_p^N \right\rfloor}} \right\rfloor = \sum_{p^k} \left\lfloor \frac{N}{p^k} \right\rfloor, 1 \le k \le \left\lfloor \log_p^N \right\rfloor$$

【例】阶乘分解

- 例如求 9! 中有多少个质因子 2：

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
包含1个2		1		1		1		1	
包含2个2				1				1	
包含3个2								1	
包含4个2	$2^k \le 9 \Rightarrow k \le \log_2{}^9 \approx 3.16 \Rightarrow$ 最多包含3个2								
2的总数	7								

- 对于每一个质数 p，只需要 $O(\log N)$ 的时间计算上述和，故整个算法的时间复杂度为 $O(N\log N)$。

练习

- 质数距离 (COGS 3478)
- 阶乘分解 (COGS 3418)
- 喜欢摇头的数学牛 (COGS 1261)
- 整数合并 (COGS 487)
- 报数 [NOIP 2021](COGS 3624)