



数论基础

ACM 中的数学小知识

陈德创
Codeforces: **dcac**

July 12, 2021

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因數

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

1 概述

- 简介
- 符号约定

2 整除与同余

- 整除
- 同余
- 快速幂

3 素数

- 素数
- 筛法

4 欧几里得

- 因数
- 最大公约数

- 最小公倍数
- 拓展欧几里得

5 欧拉函数

- 欧拉函数
- 费马小定理
- 欧拉定理

6 乘法逆元

- 简介
- 单个求解
- 多个求解

7 莫比乌斯反演

- ## ■ 数论分块

8 结束

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

主要讲什么？

- ACM 中的基础数学小知识

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

主要讲什么？

- ACM 中的基础数学小知识

有啥用？

- 单独的题目考察
- 一些题目解答的必备前提
- 玩

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

演

数论分块

结束

1. $x|y$: x 整除 y , 即 x 是 y 的因数
2. $x \bmod y$: x 除以 y 的得到的余数
3. $x \equiv y \pmod{M}$: x 同余 y (在模 M 的意义下)
4. $\gcd(x, y)$: x 和 y 的最大公约数, 简写作 (x, y)
5. $\text{lcm}(x, y)$: x 和 y 的最小公倍数, 简写作 $[x, y]$
6. $\lfloor x \rfloor$: 下取整
7. $\lceil x \rceil$: 上取整
8. Σ : 求和, 如 $\sum_{i=1}^n f(i)$, $\sum_{1 \leq i \leq n, i \in P} f(i)$

1 概述

- 简介
- 符号约定

2 整除与同余

- 整除
- 同余
- 快速幂

3 素数

- 素数
- 筛法

4 欧几里得

- 因数
- 最大公约数

- 最小公倍数
- 拓展欧几里得

5 欧拉函数

- 欧拉函数
- 费马小定理
- 欧拉定理

6 乘法逆元

- 简介
- 单个求解
- 多个求解

7 莫比乌斯反演

- ## ■ 数论分块

8 结束

整除基本性质

整除的定义：若整数 a 除以非零整数 b ，商为整数且余数为零，即 a 能被 b 整除，记做 $b|a$ ，读作： b 整除 a 或 a 能被 b 整除。 a 叫做 b 的倍数， b 叫做 a 的约数，或称为因数。

整除基本性质

整除的定义：若整数 a 除以非零整数 b ，商为整数且余数为零，即 a 能被 b 整除，记做 $b|a$ ，读作： b 整除 a 或 a 能被 b 整除。 a 叫做 b 的倍数， b 叫做 a 的约数，或称为因数。

- 若 $a|c, b|c$ ，则 $a|(b \pm c)$
- 若 $a|b$ ，则对任意的 $c(c \neq 0)$ ，有 $a|bc$
- 若 $a|b$ ，且 $b|c$ ，则 $a|c$
- 若 $a|bc$ ，且 $(a, c) = 1$ ，则 $a|b$
- 若 $c|a$ ，且 $c|b$ ，则对于任意整数 m, n ，有 $c|(ma + nb)$
- 若 $a|c, b|c$ ，且 $(a, b) = 1$ ，则 $ab|c$
- **带余除法定理：** $\forall a, b > 0, a, b \in \mathbb{N}$ ， \exists 唯一的数对 q, r ，使 $a = bq + r, (0 \leq r < b)$ 。

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

同余基本性质

- 若 $a \equiv b \pmod{m}, c \equiv d \pmod{m}$,
则 $a \pm b \equiv c \pm d \pmod{m}$
- 若 $a \equiv b \pmod{m}, c \equiv d \pmod{m}$,
则 $a \times b \equiv c \times d \pmod{m}$

同余基本性质

- 若 $a \equiv b \pmod{m}, c \equiv d \pmod{m}$,
则 $a \pm b \equiv c \pm d \pmod{m}$
- 若 $a \equiv b \pmod{m}, c \equiv d \pmod{m}$,
则 $a \times b \equiv c \times d \pmod{m}$
- 证明基本思路: $a = k_1m + b, c = k_2m + b$

同余基本性质

- 若 $a \equiv b \pmod{m}, c \equiv d \pmod{m}$,
则 $a \pm b \equiv c \pm d \pmod{m}$
- 若 $a \equiv b \pmod{m}, c \equiv d \pmod{m}$,
则 $a \times b \equiv c \times d \pmod{m}$
- 证明基本思路: $a = k_1m + b, c = k_2m + b$
- 基本用法: 求模运算下求解问题 (避免大数运算)、快速幂

同余基本性质

- 若 $a \equiv b \pmod{m}, c \equiv d \pmod{m}$,
则 $a \pm b \equiv c \pm d \pmod{m}$
- 若 $a \equiv b \pmod{m}, c \equiv d \pmod{m}$,
则 $a \times b \equiv c \times d \pmod{m}$
- 证明基本思路: $a = k_1m + b, c = k_2m + b$
- 基本用法: 求模运算下求解问题 (避免大数运算)、快速幂
- 注意: 除法不满足以上的性质

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

求解: $a^b \equiv x \pmod m, (a, b \leq 1e9)$

求解: $a^b \equiv x \pmod{m}, (a, b \leq 1e9)$

■ 朴素求法:

```
1 inline ll mpow(ll a, ll b, ll m) {  
2     ll res = 1;  
3     for (int i = 1; i <= b; ++i)  
4         res = res * a % m;  
5     return res;  
6 }
```

■ 时间复杂度: $O(b)$

求解: $a^b \equiv x \pmod{m}, (a, b \leq 1e9)$

■ 快速幂

```
1  inline ll mpow(ll a, ll b, ll m) {  
2      ll res = 1;  
3      while(b) {  
4          if (b & 1) res = res * a % m;  
5          a = a * a % m;  
6          b >>= 1;  
7      }  
8      return res;  
9  }
```

■ 时间复杂度: $O(\log b)$

■ 简介

■ 整除

■ 素数

■ 大数

■ 最大公约数

■ 拓展欧几里得

■ 欧拉函数

■ 费马小定理

■ 欧拉定理

■ 简介

■ 单个求解

■ 多个求解

■ 数论分块

8 结束

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

定义： 只能被 1 和自身整除的数称为素数，又称为质数

- 1 既不是素数，也不是合数
- 2 是最小的素数，也是唯一的偶素数

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

演

数论分块

结束

定义： 只能被 1 和自身整除的数称为素数，又称为质数

- 1 既不是素数，也不是合数
- 2 是最小的素数，也是唯一的偶素数

基本性质：

- 素数计数函数 $\pi(x)$: $\pi(x) \sim \frac{x}{\ln x}$

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

定义： 只能被 1 和自身整除的数称为素数，又称为质数

- 1 既不是素数，也不是合数
- 2 是最小的素数，也是唯一的偶素数

基本性质：

- 素数计数函数 $\pi(x)$: $\pi(x) \sim \frac{x}{\ln x}$
- 素数的分布: $1e9$ 范围内，任意两个相邻的素数差不超过 400

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

定义： 只能被 1 和自身整除的数称为素数，又称为质数

- 1 既不是素数，也不是合数
- 2 是最小的素数，也是唯一的偶素数

基本性质：

- 素数计数函数 $\pi(x)$: $\pi(x) \sim \frac{x}{\ln x}$
- 素数的分布: $1e9$ 范围内，任意两个相邻的素数差不超过 400
- 威尔逊定理: $(p-1)! \equiv -1 \pmod{p}$

素数的判定

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

演

数论分块

结束

- 暴力枚举所有可能的因子即可
- 事实：如果 $x|a$ ，那么 $\frac{a}{x}|a$ 。
- 判定算法：

```
1 inline bool isPrime(ll n) {  
2     if (n < 2) return false;  
3     for (int i = 2; i <= sqrt(n) + 1; ++i) {  
4         if (!(n % i)) return false;  
5     }  
6     return true;  
7 }
```

- 时间复杂度： $O(\sqrt{n})$

素数的判定

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

莫比乌斯反演

数论分块

结束

- 暴力枚举所有可能的因子即可
- 事实：如果 $x|a$ ，那么 $\frac{a}{x}|a$ 。
- 判定算法：

```

1 inline bool isPrime(ll n) {
2     if (n < 2) return false;
3     for (int i = 2; i <= sqrt(n) + 1; ++i) {
4         if (!(n % i)) return false;
5     }
6     return true;
7 }

```

- 时间复杂度： $O(\sqrt{n})$
- Miller-Rabin 素性测试： $O(k \log^3 n)$
- 我不会

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

筛法用于求出 $1 \sim n$ 中所有的素数。

- 事实：一个数 x 的任意整数倍一定是素数

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

筛法用于求出 $1 \sim n$ 中所有的素数。

- 事实：一个数 x 的任意整数倍一定是素数
- 对每个数，将其倍速标记为非素数，那么没被标记的数，就是素数

筛法用于求出 $1 \sim n$ 中所有的素数。

- 事实：一个数 x 的任意整数倍一定是素数
- 对每个数，将其倍速标记为非素数，那么没被标记的数，就是素数
- 算法实现：

```
1  const int MAXN = 1e6 + 5;  
2  bool inp[MAXN], prime[MAXN], cnt;  
3  inline void getPrime(int n) {  
4      memset(inp, 0, sizeof(inp));  
5      for (int i = 2; i <= n; ++i) {  
6          if (!inp[i]) prime[++cnt] = i;  
7          for (int j = 2; i * j <= n; ++j)  
8              inp[i * j] = true;  
9      }  
10 }
```

- 时间复杂度： $O(n \log n)$

埃拉托斯特尼筛法

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

- 只需要对素数的倍数进行标记就好了

埃拉托斯特尼筛法

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

演

数论分块

结束

- 只需要对素数的倍数进行标记就好了
- 算法实现：

```
1 inline void getPrime(int n) {  
2     memset(inp, 0, sizeof(inp));  
3     for (int i = 2; i <= n; ++i) {  
4         if (inp[i]) continue;  
5         prime[++cnt] = i;  
6         for (int j = 2; i * j <= n; ++j)  
7             inp[i * j] = true;  
8     }  
9 }
```

- 时间复杂度： $O(n \log \log n)$

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

- 如果能让每个合数都只被标记一次就好了

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

- 如果能让每个合数都只被标记一次就好了
- 算法实现：

```
1 inline void getPrime(int n) {  
2     for (int i = 2; i <= n; ++i) {  
3         if (!inp[i]) prime[++cnt] = i;  
4         for (int j = 1; j <= cnt && i * prime[j] <= n; ++j) {  
5             inp[i * prime[j]] = true;  
6             if (!(i % prime[j])) break;  
7         }  
8     }  
9 }
```

- 如果能让每个合数都只被标记一次就好了
- 算法实现：

```
1 inline void getPrime(int n) {  
2     for (int i = 2; i <= n; ++i) {  
3         if (!inp[i]) prime[++cnt] = i;  
4         for (int j = 1; j <= cnt && i * prime[j] <= n; ++j) {  
5             inp[i * prime[j]] = true;  
6             if (!(i % prime[j])) break;  
7         }  
8     }  
9 }
```

- 时间复杂度： $O(n)$
- 非常重要，可以用来线性求解某些积性函数

1 概述

- 简介
- 符号约定

2 整除与同余

- 整除
- 同余
- 快速幂

3 素数

- 素数
- 筛法

4 欧几里得

- 因数
- 最大公约数

- 最小公倍数
- 拓展欧几里得

5 欧拉函数

- 欧拉函数
- 费马小定理
- 欧拉定理

6 乘法逆元

- 简介
- 单个求解
- 多个求解

7 莫比乌斯反演

- ## ■ 数论分块

8 结束

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

定理 (算术基本定理)

$\forall A \in \mathbb{N}, A > 1 \quad \exists p_1 < p_2 < p_3 < \cdots < p_n, a_i \in \mathbb{Z}^+ \text{ s.t. } \prod_{i=1}^n p_i^{a_i} = A$
其中 p_i 是一个质数。这种表示的方法存在，而且是唯一的。

定理 (算术基本定理)

$\forall A \in \mathbb{N}, A > 1 \quad \exists p_1 < p_2 < p_3 < \cdots < p_n, a_i \in \mathbb{Z}^+ \text{ s.t. } \prod_{i=1}^n p_i^{a_i} = A$
其中 p_i 是一个质数。这种表示的方法存在，而且是唯一的。

- $1 \sim 1e9$ 的数最多有 1344 个因子。(by 欢神)

定理 (算术基本定理)

$\forall A \in \mathbb{N}, A > 1 \quad \exists p_1 < p_2 < p_3 < \dots < p_n, a_i \in \mathbb{Z}^+ \text{ s.t. } \prod_{i=1}^n p_i^{a_i} = A$
 其中 p_i 是一个质数。这种表示的方法存在，而且是唯一的。

- $1 \sim 1e9$ 的数最多有 1344 个因子。(by 欢神)
- 求解一个数的所有因数：

```

1  vector<int> breakdown(int n) {
2      vector<int> res;
3      for (int i = 2; i * i <= n; i++) {
4          if (n % i == 0) {
5              res.push_back(i);
6              res.push_back(n / i);
7          }
8      }
9      return res;
10 }
```

- 时间复杂度： $O(\sqrt{n})$

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

- 求解一个数的所有质因数（即质因数分解）：

■ 求解一个数的所有质因数（即质因数分解）：

```

1 vector<int> breakdown(int n) {
2     vector<int> res;
3     for (int i = 2; i * i <= n; i++) {
4         if (n % i == 0) {
5             while (n % i == 0) n /= i;
6             res.push_back(i);
7         }
8     }
9     if (n != 1) res.push_back(n);
10    return res;
11 }

```

■ 时间复杂度： $O(\sqrt{\frac{n}{\ln n}})$

■ 求解一个数的所有质因数（即质因数分解）：

```

1 vector<int> breakdown(int n) {
2     vector<int> res;
3     for (int i = 2; i * i <= n; i++) {
4         if (n % i == 0) {
5             while (n % i == 0) n /= i;
6             res.push_back(i);
7         }
8     }
9     if (n != 1) res.push_back(n);
10    return res;
11 }

```

■ 时间复杂度： $O(\sqrt{\frac{n}{\ln n}})$

■ 详细介绍：分解质因数 - OI-Wiki

最大公约数

GCD

最大公约数，是指一组数所有公约数里面最大的一个。

最大公约数

GCD

最大公约数，是指一组数所有公约数里面最大的一个。

■ a 和 b 的最大公约数记为： $\gcd(a, b)$

最大公约数

GCD

最大公约数，是指一组数所有公约数里面最大的一个。

■ a 和 b 的最大公约数记为： $\gcd(a, b)$

■ 事实：

$$\gcd(a, b) = \gcd(b, a - b)$$

$$\gcd(a, b) = \gcd(a, a \bmod b)$$

最大公约数

GCD

最大公约数，是指一组数所有公约数里面最大的一个。

■ a 和 b 的最大公约数记为： $\gcd(a, b)$

■ **事实：**

$$\gcd(a, b) = \gcd(b, a - b)$$

$$\gcd(a, b) = \gcd(a, a \bmod b)$$

■ 集合 $S = \{\gcd(a_i, \dots, a_n) | 1 \leq i \leq n\}$ 至多有 $O(\max(a_i))$ 个元素。

欧几里得算法

■ 算法实现：

```
1 int gcd(int a, int b) {  
2     if (b == 0) return a;  
3     return gcd(b, a % b);  
4 }
```

■ 时间复杂度： $O(\log n)$

欧几里得算法

■ 算法实现：

```
1 int gcd(int a, int b) {  
2     if (b == 0) return a;  
3     return gcd(b, a % b);  
4 }
```

■ 时间复杂度： $O(\log n)$

■ 求解斐波那契数列相邻两项时达到最坏时间复杂度

欧几里得算法

■ 算法实现：

```
1 int gcd(int a, int b) {  
2     if (b == 0) return a;  
3     return gcd(b, a % b);  
4 }
```

■ 时间复杂度： $O(\log n)$

■ 求解斐波那契数列相邻两项时达到最坏时间复杂度

■ `<algorithm>`库内置了`__gcd()`函数

欧几里得算法

■ 算法实现：

```
1 int gcd(int a, int b) {  
2     if (b == 0) return a;  
3     return gcd(b, a % b);  
4 }
```

- 时间复杂度： $O(\log n)$
- 求解斐波那契数列相邻两项时达到最坏时间复杂度
- `<algorithm>`库内置了`__gcd()`函数
- 本质上是将大规模问题转化为小规模问题
- 将问题 (a, b) 转化为了 $(b, a \bmod b)$

多个数的最大公约数

求解多个数的最大公约数：

- 可以证明， $\gcd(a, \gcd(b, c)) = \gcd(a, b, c)$
- 即每次取出两个数求出 \gcd 后再放 \gcd 回去，不会对所需要的答案造成影响。

多个数的最大公约数

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

求解多个数的最大公约数：

- 可以证明, $\gcd(a, \gcd(b, c)) = \gcd(a, b, c)$
- 即每次取出两个数求出 \gcd 后再放 \gcd 回去, 不会对所需要的答案造成影响。

```
1 int gcd(int a[], int n) {  
2     int res = a[1];  
3     for (int i = 2; i <= n; ++i)  
4         res = __gcd(res, a[i]);  
5     return res;  
6 }
```

例题

又是毕业季

题目链接：[又是毕业季 I](#)

题目描述

给定 n 和 k ，问在 $1 \sim n$ 中选 k 个数使得它们 \gcd 最大，求最大值是多少。
 $1 \leq k \leq n \leq 1e9$

例题

又是毕业季

题目链接：[又是毕业季 I](#)

题目描述

给定 n 和 k ，问在 $1 \sim n$ 中选 k 个数使得它们 \gcd 最大，求最大值是多少。
 $1 \leq k \leq n \leq 1e9$

题解

- 反向考虑，即考虑一个数 x ， $1 \sim n$ 中有多少数被它整除。

例题

又是毕业季

题目链接: [又是毕业季 I](#)

题目描述

给定 n 和 k , 问在 $1 \sim n$ 中选 k 个数使得它们 \gcd 最大, 求最大值是多少。
 $1 \leq k \leq n \leq 1e9$

题解

- 反向考虑, 即考虑一个数 x , $1 \sim n$ 中有多少数被它整除。
- 显然为 $\lfloor \frac{n}{x} \rfloor$ 个
- 即求 $\max_x \{ \lfloor \frac{n}{x} \rfloor \geq k \}$, 显然答案为 $\lfloor \frac{n}{k} \rfloor$

例题

又是毕业季

题目链接: [又是毕业季 I](#)

题目描述

给定 n 和 k , 问在 $1 \sim n$ 中选 k 个数使得它们 \gcd 最大, 求最大值是多少。
 $1 \leq k \leq n \leq 1e9$

题解

- 反向考虑, 即考虑一个数 x , $1 \sim n$ 中有多少数被它整除。
- 显然为 $\lfloor \frac{n}{x} \rfloor$ 个
- 即求 $\max_x \{ \lfloor \frac{n}{x} \rfloor \geq k \}$, 显然答案为 $\lfloor \frac{n}{k} \rfloor$
- 时间复杂度: $O(1)$

例题

又是毕业季

题目链接：[又是毕业季 I](#)

题目描述

给定 n 和 k ，问在 $1 \sim n$ 中选 k 个数使得它们 \gcd 最大，求最大值是多少。
 $1 \leq k \leq n \leq 1e9$

题解

- 反向考虑，即考虑一个数 x ， $1 \sim n$ 中有多少数被它整除。
- 显然为 $\lfloor \frac{n}{x} \rfloor$ 个
- 即求 $\max_x \{ \lfloor \frac{n}{x} \rfloor \geq k \}$ ，显然答案为 $\lfloor \frac{n}{k} \rfloor$
- 时间复杂度： $O(1)$

```
1 n, k = map(int, input().split())
2 print(int(n / k))
```

例题

又是毕业季

题目链接：[又是毕业季 I](#)

题目描述

给定 n 和 k ，问在 $1 \sim n$ 中选 k 个数使得它们 gcd 最大，求最大值是多少。
 $1 \leq k \leq n \leq 1e9$

题解

- 反向考虑，即考虑一个数 x ， $1 \sim n$ 中有多少数被它整除。
- 显然为 $\lfloor \frac{n}{x} \rfloor$ 个
- 即求 $\max_x \{ \lfloor \frac{n}{x} \rfloor \geq k \}$ ，显然答案为 $\lfloor \frac{n}{k} \rfloor$
- 时间复杂度： $O(1)$

```
1 n, k = map(int, input().split())
2 print(int(n / k))
```

同样思想解决的题：[又是毕业季 II](#)（双倍经验它不香吗？）。

例题

最大公约数

题目链接：[最大公约数](#)

题目描述

有一个 $n \times m$ 的矩阵 a 。对此矩阵进行变换，定义为将这个矩阵内的所有元素变为其上下左右四个元素（不存在则忽略）及自身的最大公约数。

询问 $a_{x,y}$ 在进行最少多少次变换之后会变成 1。如果可以使 $a_{x,y}$ 经过若干次变换变成 1，输出其中最少的次数；否则输出 -1 。

例题

最大公约数

题目链接：[最大公约数](#)

题目描述

有一个 $n \times m$ 的矩阵 a 。对此矩阵进行变换，定义为将这个矩阵内的所有元素变为其上下左右四个元素（不存在则忽略）及自身的最大公约数。

询问 $a_{x,y}$ 在进行最少多少次变换之后会变成 1。如果可以使 $a_{x,y}$ 经过若干次变换变成 1，输出其中最少的次数；否则输出 -1。

数据范围：

$$1 \leq n, m \leq 2 \times 10^3, 1 \leq a_{i,j} \leq 10^{18}, 1 \leq x \leq n, 1 \leq y \leq m$$

例题

最大公约数

题目链接：[最大公约数](#)

题目描述

有一个 $n \times m$ 的矩阵 a 。对此矩阵进行变换，定义为将这个矩阵内的所有元素变为其上下左右四个元素（不存在则忽略）及自身的最大公约数。

询问 $a_{x,y}$ 在进行最少多少次变换之后会变成 1。如果可以使 $a_{x,y}$ 经过若干次变换变成 1，输出其中最少的次数；否则输出 -1。

数据范围：

$$1 \leq n, m \leq 2 \times 10^3, 1 \leq a_{i,j} \leq 10^{18}, 1 \leq x \leq n, 1 \leq y \leq m$$

hint

■ 事实：

$$\gcd(a, \gcd(b, c)) = \gcd(a, b, c)$$

$$\gcd(\gcd(a, b), \gcd(b, c)) = \gcd(a, b, c)$$

■ 分别考虑“判断可能”和“求解”。

例题

最大公约数

题解

显然，当且仅当所有元素的 \gcd 不为 1 时，答案不存在。

例题

最大公约数

题解

显然，当且仅当所有元素的 \gcd 不为 1 时，答案不存在。
当答案存在时，我们考虑每一次矩阵变换。

例题

最大公约数

题解

显然，当且仅当所有元素的 \gcd 不为 1 时，答案不存在。
当答案存在时，我们考虑每一次矩阵变换。

- 第一次， $a_{x,y}$ 由 $a_{x,y}$ 上下左右四个点来更新。
即 $a_{x,y} = \gcd(a_{x-1,y}, a_{x+1,y}, a_{x,y-1}, a_{x,y+2})$ 。
- 记 $a_{x,y}$ 周围的四个点为 b_1, b_2, b_3, b_4 。
同时 b_i 也由它们的上下左右四个点来更新

例题

最大公约数

题解

显然，当且仅当所有元素的 \gcd 不为 1 时，答案不存在。
当答案存在时，我们考虑每一次矩阵变换。

- 第一次， $a_{x,y}$ 由 $a_{x,y}$ 上下左右四个点来更新。
即 $a_{x,y} = \gcd(a_{x-1,y}, a_{x+1,y}, a_{x,y-1}, a_{x,y+2})$ 。
- 记 $a_{x,y}$ 周围的四个点为 b_1, b_2, b_3, b_4 。
同时 b_i 也由它们的上下左右四个点来更新
- 第二次，相当于 $a_{x,y}$ 由每个 b_i 周围的四个点更新。
即相当于 $a_{x,y}$ 由距离它小于 2 的点更新。
- 同时 b_i 也是由距离它小于 2 的点更新。

例题

最大公约数

题解

显然，当且仅当所有元素的 \gcd 不为 1 时，答案不存在。
当答案存在时，我们考虑每一次矩阵变换。

- 第一次， $a_{x,y}$ 由 $a_{x,y}$ 上下左右四个点来更新。
即 $a_{x,y} = \gcd(a_{x-1,y}, a_{x+1,y}, a_{x,y-1}, a_{x,y+2})$ 。
- 记 $a_{x,y}$ 周围的四个点为 b_1, b_2, b_3, b_4 。
同时 b_i 也由它们的上下左右四个点来更新
- 第二次，相当于 $a_{x,y}$ 由每个 b_i 周围的四个点更新。
即相当于 $a_{x,y}$ 由距离它小于 2 的点更新。
- 同时 b_i 也是由距离它小于 2 的点更新。
- 以此类推，假设第 k 次， $a_{x,y}$ 由距离它小于 k 的点更新。
那么 b_i 也是由距离它小于 k 的点更新。

例题

最大公约数

题解

显然，当且仅当所有元素的 \gcd 不为 1 时，答案不存在。
当答案存在时，我们考虑每一次矩阵变换。

- 第一次， $a_{x,y}$ 由 $a_{x,y}$ 上下左右四个点来更新。
即 $a_{x,y} = \gcd(a_{x-1,y}, a_{x+1,y}, a_{x,y-1}, a_{x,y+2})$ 。
- 记 $a_{x,y}$ 周围的四个点为 b_1, b_2, b_3, b_4 。
同时 b_i 也由它们的上下左右四个点来更新
- 第二次，相当于 $a_{x,y}$ 由每个 b_i 周围的四个点更新。
即相当于 $a_{x,y}$ 由距离它小于 2 的点更新。
- 同时 b_i 也是由距离它小于 2 的点更新。
- 以此类推，假设第 k 次， $a_{x,y}$ 由距离它小于 k 的点更新。
那么 b_i 也是由距离它小于 k 的点更新。
- 那么显然第 $k+1$ 次， $a_{x,y}$ 由距离 b_i 小于 k 的点更新。
即相当于距离它小于 $k+1$ 的点更新。

例题

最大公约数

题解

- 即我们归纳证明了，第 k 次， $a_{x,y}$ 由距离它小于 k 的点更新。
- 即，第 k 次变换后， $a_{x,y}$ 的值变成了所有距它小于等于 k 的点的值的 \gcd 。

例题

最大公约数

题解

- 即我们归纳证明了，第 k 次， $a_{x,y}$ 由距离它小于 k 的点更新。
- 即，第 k 次变换后， $a_{x,y}$ 的值变成了所有距它小于等于 k 的点的值的 \gcd 。
- 题目转化为：从 (x,y) 开始一层层取 \gcd ，直到 $\gcd = 1$ 。

例题

最大公约数

题解

- 即我们归纳证明了，第 k 次， $a_{x,y}$ 由距离它小于 k 的点更新。
- 即，第 k 次变换后， $a_{x,y}$ 的值变成了所有距它小于等于 k 的点的值的 \gcd 。
- 题目转化为：从 (x,y) 开始一层层取 \gcd ，直到 $\gcd = 1$ 。
- 从 (x,y) 开始 bfs，同时对每个遍历到的元素取 \gcd ，直到 $\gcd = 1$ 。
- 遍历层数就是答案。

最小公倍数

LCM

■ 事实：

$$\forall a, b > 0, a, b \in \mathbb{N}, a \times b = \gcd(a, b) \times \text{lcm}(a, b)$$

■ 所以求出俩数的 GCD 后 LCM 可在 $O(1)$ 的时间内求出

最小公倍数

LCM

■ 事实：

$$\forall a, b > 0, a, b \in \mathbb{N}, a \times b = \gcd(a, b) \times \text{lcm}(a, b)$$

■ 所以求出俩数的 GCD 后 LCM 可在 $O(1)$ 的时间内求出

■ 算法实现：

```
1 int lcm(int a, int b) {  
2     return a * b / __gcd(a, b);  
3 }
```

最小公倍数

LCM

■ 事实：

$$\forall a, b > 0, a, b \in \mathbb{N}, a \times b = \gcd(a, b) \times \text{lcm}(a, b)$$

■ 所以求出俩数的 GCD 后 LCM 可在 $O(1)$ 的时间内求出

■ 算法实现：

```
1 int lcm(int a, int b) {  
2     return a * b / __gcd(a, b);  
3 }
```

■ 多个数的 LCM

■ 与多个数的 GCD 类似

最小公倍数

LCM

■ 事实：

$$\forall a, b > 0, a, b \in \mathbb{N}, a \times b = \gcd(a, b) \times \text{lcm}(a, b)$$

■ 所以求出俩数的 GCD 后 LCM 可在 $O(1)$ 的时间内求出

■ 算法实现：

```
1 int lcm(int a, int b) {  
2     return a * b / __gcd(a, b);  
3 }
```

■ 多个数的 LCM

- 与多个数的 GCD 类似
- 每次取出两个数求 LCM 后再将 LCM 放回去，不会对所需要的答案造成影响

例题

最大公约数和最小公倍数问题

题目链接: [NOIP2001 普及组] 最大公约数和最小公倍数问题

题目描述

输入两个正整数 x_0, y_0 , ($2 \leq x_0, y_0 \leq 10^5$), 求出满足下列条件的 P, Q 的个数:

- P, Q 是正整数。
- 要求 P, Q 以 x_0 为最大公约数, 以 y_0 为最小公倍数。

试求: 满足条件的所有可能的 P, Q 的个数。

例题

最大公约数和最小公倍数问题

题目链接: [NOIP2001 普及组] 最大公约数和最小公倍数问题

题目描述

输入两个正整数 x_0, y_0 , ($2 \leq x_0, y_0 \leq 10^5$), 求出满足下列条件的 P, Q 的个数:

- P, Q 是正整数。
- 要求 P, Q 以 x_0 为最大公约数, 以 y_0 为最小公倍数。

试求: 满足条件的所有可能的 P, Q 的个数。

题解

- 根据 $P \times Q = \gcd(P, Q) \times \text{lcm}(P, Q) = x_0 y_0$, 我们可以得到 $P \times Q$ 的值
- 枚举 $i | (x_0 y_0)$, 考虑 $i, \lfloor \frac{x_0 y_0}{i} \rfloor$ 的最大公约数和最小公倍数是不是分别为 P, Q 即可

例题

最大公约数和最小公倍数问题

题目链接: [NOIP2001 普及组] 最大公约数和最小公倍数问题

题目描述

输入两个正整数 x_0, y_0 , ($2 \leq x_0, y_0 \leq 10^5$), 求出满足下列条件的 P, Q 的个数:

- P, Q 是正整数。
- 要求 P, Q 以 x_0 为最大公约数, 以 y_0 为最小公倍数。

试求: 满足条件的所有可能的 P, Q 的个数。

题解

- 根据 $P \times Q = \gcd(P, Q) \times \text{lcm}(P, Q) = x_0 y_0$, 我们可以得到 $P \times Q$ 的值
- 枚举 $i | (x_0 y_0)$, 考虑 $i, \lfloor \frac{x_0 y_0}{i} \rfloor$ 的最大公约数和最小公倍数是不是分别为 P, Q 即可
- 时间复杂度: $O(n \log n)$

拓展欧几里得算法

简介

- 拓展欧几里得算法用于求解 $ax + by = \gcd(a, b)$ 的一组可行解

拓展欧几里得算法

简介

- 拓展欧几里得算法用于求解 $ax + by = \gcd(a, b)$ 的一组可行解
- 常用于求解**乘法逆元**

拓展欧几里得算法

简介

- 拓展欧几里得算法用于求解 $ax + by = \gcd(a, b)$ 的一组可行解
- 常用于求解**乘法逆元**
- **裴蜀定理**：该方程组的解一定存在

定理（裴蜀定理）

设 a, b 是不全为零的整数，则存在整数 x, y ，使得 $ax + by = \gcd(a, b)$ 。

拓展欧几里得算法

简介

- 拓展欧几里得算法用于求解 $ax + by = \gcd(a, b)$ 的一组可行解
- 常用于求解乘法逆元
- 裴蜀定理：该方程组的解一定存在

定理（裴蜀定理）

设 a, b 是不全为零的整数，则存在整数 x, y ，使得 $ax + by = \gcd(a, b)$ 。

- 算法思想：将问题 (a, b) 转化为 $(b, a \bmod b)$

拓展欧几里得算法

简介

- 拓展欧几里得算法用于求解 $ax + by = \gcd(a, b)$ 的一组可行解
- 常用于求解乘法逆元
- 裴蜀定理：该方程组的解一定存在

定理（裴蜀定理）

设 a, b 是不全为零的整数，则存在整数 x, y ，使得 $ax + by = \gcd(a, b)$ 。

- 算法思想：将问题 (a, b) 转化为 $(b, a \bmod b)$
- 要注意负值

拓展欧几里得算法

证明

Proof.

拓展欧几里得算法

证明

Proof.

$$\begin{aligned} ax_1 + by_1 &= \gcd(a, b) \\ bx_2 + (a \bmod b)y_2 &= \gcd(b, a \bmod b) \end{aligned}$$

拓展欧几里得算法

证明

Proof.

$$ax_1 + by_1 = \gcd(a, b)$$

$$bx_2 + (a \bmod b)y_2 = \gcd(b, a \bmod b)$$

$$\therefore \gcd(a, b) = \gcd(b, a \bmod b)$$

拓展欧几里得算法

证明

Proof.

$$ax_1 + by_1 = \gcd(a, b)$$

$$bx_2 + (a \bmod b)y_2 = \gcd(b, a \bmod b)$$

$$\because \gcd(a, b) = \gcd(b, a \bmod b)$$

$$\therefore ax_1 + by_1 = bx_2 + (a \bmod b)y_2$$

拓展欧几里得算法

证明

Proof.

$$\begin{aligned}
 ax_1 + by_1 &= \gcd(a, b) \\
 bx_2 + (a \bmod b)y_2 &= \gcd(b, a \bmod b) \\
 \therefore \gcd(a, b) &= \gcd(b, a \bmod b) \\
 \therefore ax_1 + by_1 &= bx_2 + (a \bmod b)y_2 \\
 \therefore a \bmod b &= a - \left(\left\lfloor \frac{a}{b} \right\rfloor \times b\right)
 \end{aligned}$$

拓展欧几里得算法

证明

Proof.

$$\begin{aligned}
 ax_1 + by_1 &= \gcd(a, b) \\
 bx_2 + (a \bmod b)y_2 &= \gcd(b, a \bmod b) \\
 \therefore \gcd(a, b) &= \gcd(b, a \bmod b) \\
 \therefore ax_1 + by_1 &= bx_2 + (a \bmod b)y_2 \\
 \therefore a \bmod b &= a - \left(\left\lfloor \frac{a}{b} \right\rfloor \times b\right) \\
 \therefore ax_1 + by_1 &= bx_2 + \left(a - \left(\left\lfloor \frac{a}{b} \right\rfloor \cdot b\right)\right)y_2 \\
 &= ay_2 + bx_2 - \left\lfloor \frac{a}{b} \right\rfloor \cdot by_2
 \end{aligned}$$

拓展欧几里得算法

证明

Proof.

$$\begin{aligned}
 ax_1 + by_1 &= \gcd(a, b) \\
 bx_2 + (a \bmod b)y_2 &= \gcd(b, a \bmod b) \\
 \therefore \gcd(a, b) &= \gcd(b, a \bmod b) \\
 \therefore ax_1 + by_1 &= bx_2 + (a \bmod b)y_2 \\
 \therefore a \bmod b &= a - \left(\left\lfloor \frac{a}{b} \right\rfloor \times b\right) \\
 \therefore ax_1 + by_1 &= bx_2 + \left(a - \left(\left\lfloor \frac{a}{b} \right\rfloor \cdot b\right)\right)y_2 \\
 &= ay_2 + bx_2 - \left\lfloor \frac{a}{b} \right\rfloor \cdot by_2 \\
 \therefore ax_1 + by_1 &= ay_2 + b\left(x_2 - \left\lfloor \frac{a}{b} \right\rfloor y_2\right)
 \end{aligned}$$

拓展欧几里得算法

证明

Proof.

$$\begin{aligned}
 ax_1 + by_1 &= \gcd(a, b) \\
 bx_2 + (a \bmod b)y_2 &= \gcd(b, a \bmod b) \\
 \therefore \gcd(a, b) &= \gcd(b, a \bmod b) \\
 \therefore ax_1 + by_1 &= bx_2 + (a \bmod b)y_2 \\
 \therefore a \bmod b &= a - \left(\left\lfloor \frac{a}{b} \right\rfloor \times b\right) \\
 \therefore ax_1 + by_1 &= bx_2 + \left(a - \left(\left\lfloor \frac{a}{b} \right\rfloor \cdot b\right)\right)y_2 \\
 &= ay_2 + bx_2 - \left\lfloor \frac{a}{b} \right\rfloor \cdot by_2 \\
 \therefore ax_1 + by_1 &= ay_2 + b\left(x_2 - \left\lfloor \frac{a}{b} \right\rfloor y_2\right) \\
 \therefore a &= a, \quad b = b \\
 \therefore x_1 &= y_2, \quad y_1 = x_2 - \left\lfloor \frac{a}{b} \right\rfloor y_2
 \end{aligned}$$



拓展欧几里得算法

实现

- $x_1 = y_2, y_1 = x_2 - \lfloor \frac{a}{b} \rfloor y_2$
- 其中 x_1, y_1 是问题 (a, b) 的答案
 x_2, y_2 是问题 $(b, a \bmod b)$ 的答案

拓展欧几里得算法

实现

- $x_1 = y_2, y_1 = x_2 - \lfloor \frac{a}{b} \rfloor y_2$
- 其中 x_1, y_1 是问题 (a, b) 的答案
 x_2, y_2 是问题 $(b, a \bmod b)$ 的答案
- 算法实现:

```
1  ll exgcd(ll a, ll b, ll& x, ll& y){
2      if (!b){
3          x = 1, y = 0;
4          return a;
5      }
6      ll gcd = exgcd(b, a % b, y, x);
7      y -= a / b * x;
8      return gcd;
9  }
```

拓展欧几里得算法

拓展

- 我们已经知道了不定方程 $ax + by = c$ 的一组解为 (x_0, y_0) 那么它的全部解为:

$$\begin{cases} x = x_0 + \frac{b}{\gcd(a,b)} * t \\ y = y_0 - \frac{a}{\gcd(a,b)} * t \end{cases}, t \in \mathbb{Z}$$

拓展欧几里得算法

拓展

- 我们已经知道了不定方程 $ax + by = c$ 的一组解为 (x_0, y_0) 那么它的全部解为:

$$\begin{cases} x = x_0 + \frac{b}{\gcd(a,b)} * t \\ y = y_0 - \frac{a}{\gcd(a,b)} * t \end{cases}, t \in \mathbb{Z}$$

- 另外，拓欧算法中，以下定理成立：

定理

当 $a, b > 0$ 且 $\gcd(a, b) \neq \min(a, b)$ 时有 $|x| < \frac{b}{\gcd(a, b)}$ 和 $|y| < \frac{a}{\gcd(a, b)}$

拓展欧几里得算法

拓展

- 我们已经知道了不定方程 $ax + by = c$ 的一组解为 (x_0, y_0) 那么它的全部解为:

$$\begin{cases} x = x_0 + \frac{b}{\gcd(a,b)} * t \\ y = y_0 - \frac{a}{\gcd(a,b)} * t \end{cases}, t \in \mathbb{Z}$$

- 另外，拓欧算法中，以下定理成立：

定理

当 $a, b > 0$ 且 $\gcd(a, b) \neq \min(a, b)$ 时有 $|x| < \frac{b}{\gcd(a,b)}$ 和 $|y| < \frac{a}{\gcd(a,b)}$

- 此定理告知我们以下两个事实：

1. 扩欧过程中不会发生溢出
2. 不定方程的某个最小正整数解为 $\left(x + \frac{b}{\gcd(a,b)}\right) \bmod \frac{b}{\gcd(a,b)}$
和 $\left(y + \frac{a}{\gcd(a,b)}\right) \bmod \frac{a}{\gcd(a,b)}。$

例题

同余方程

题目链接: [NOIP2012 提高组] 同余方程

题目描述

求关于 x 的同余方程 $ax \equiv 1 \pmod{b}$ 的最小正整数解。

数据保证一定有解。

$2 \leq a, b \leq 2e9$

例题

同余方程

题目链接: [NOIP2012 提高组] 同余方程

题目描述

求关于 x 的同余方程 $ax \equiv 1 \pmod{b}$ 的最小正整数解。

数据保证一定有解。

$2 \leq a, b \leq 2e9$

题解

- $ax \equiv 1 \pmod{b}$ 可以转化为 $ax = (-b)y + 1$, 即 $ax + by = 1$ 。

例题

同余方程

题目链接: [NOIP2012 提高组] 同余方程

题目描述

求关于 x 的同余方程 $ax \equiv 1 \pmod{b}$ 的最小正整数解。

数据保证一定有解。

$2 \leq a, b \leq 2e9$

题解

- $ax \equiv 1 \pmod{b}$ 可以转化为 $ax = (-b)y + 1$, 即 $ax + by = 1$ 。
- 拓展欧几里得裸题。
- 注意判断负值。

例题

同余方程

题目链接: [NOIP2012 提高组] 同余方程

题目描述

求关于 x 的同余方程 $ax \equiv 1 \pmod{b}$ 的最小正整数解。

数据保证一定有解。

$2 \leq a, b \leq 2e9$

题解

- $ax \equiv 1 \pmod{b}$ 可以转化为 $ax = (-b)y + 1$, 即 $ax + by = 1$ 。
- 拓展欧几里得裸题。
- 注意判断负值。
- 时间复杂度: $O(\log a)$

例题

同余方程

题目链接: [NOIP2012 提高组] 同余方程

题目描述

求关于 x 的同余方程 $ax \equiv 1 \pmod{b}$ 的最小正整数解。

数据保证一定有解。

$2 \leq a, b \leq 2e9$

题解

- $ax \equiv 1 \pmod{b}$ 可以转化为 $ax = (-b)y + 1$, 即 $ax + by = 1$ 。
- 拓展欧几里得裸题。
- 注意判断负值。
- 时间复杂度: $O(\log a)$

拓展欧几里得习题推荐:

- P1516 青蛙的约会 - 洛谷
- 5512 Pagodas - HDOJ (裴蜀定理)

欧拉函数

基本概念

定义：欧拉函数，即 $\varphi(n)$ ，表示小于等于 n 且和 n 互质的数的个数。特别地， $\varphi(1) = 1$ 。

$$\varphi(n) = \sum_{i=1}^n [\gcd(i, n) = 1]$$

显然，当 n 是质数时， $\varphi(n) = n - 1$ 。

欧拉函数

基本概念

定义：欧拉函数，即 $\varphi(n)$ ，表示小于等于 n 且和 n 互质的数的个数。特别地， $\varphi(1) = 1$ 。

$$\varphi(n) = \sum_{i=1}^n [\gcd(i, n) = 1]$$

显然，当 n 是质数时， $\varphi(n) = n - 1$ 。

性质：

- 欧拉函数是积性函数（证明：[欧拉函数的积性证明](#)）
 - **积性函数：**若 $f(n)$ 是积性函数，那么当 $\gcd(a, b) = 1$ 时，有 $f(a \cdot b) = f(a) \cdot f(b)$ 。
 - **完全积性函数：**若 $f(n)$ 是完全积性函数，那么有 $f(a \cdot b) = f(a) \cdot f(b)$

欧拉函数

基本概念

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

反演

数论分块

结束

定义：欧拉函数，即 $\varphi(n)$ ，表示小于等于 n 且和 n 互质的数的个数。特别地， $\varphi(1) = 1$ 。

$$\varphi(n) = \sum_{i=1}^n [\gcd(i, n) = 1]$$

显然，当 n 是质数时， $\varphi(n) = n - 1$ 。

性质：

- 欧拉函数是积性函数（证明：[欧拉函数的积性证明](#)）
 - **积性函数：**若 $f(n)$ 是积性函数，那么当 $\gcd(a, b) = 1$ 时，有 $f(a \cdot b) = f(a) \cdot f(b)$ 。
 - **完全积性函数：**若 $f(n)$ 是完全积性函数，那么有 $f(a \cdot b) = f(a) \cdot f(b)$
- $n = \sum_{d|n} \varphi(d)$ 。由莫比乌斯反演可得。

欧拉函数

基本概念

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

反演

数论分块

结束

定义：欧拉函数，即 $\varphi(n)$ ，表示小于等于 n 且和 n 互质的数的个数。特别地， $\varphi(1) = 1$ 。

$$\varphi(n) = \sum_{i=1}^n [\gcd(i, n) = 1]$$

显然，当 n 是质数时， $\varphi(n) = n - 1$ 。

性质：

- 欧拉函数是积性函数（证明：[欧拉函数的积性证明](#)）
 - **积性函数：**若 $f(n)$ 是积性函数，那么当 $\gcd(a, b) = 1$ 时，有 $f(a \cdot b) = f(a) \cdot f(b)$ 。
 - **完全积性函数：**若 $f(n)$ 是完全积性函数，那么有 $f(a \cdot b) = f(a) \cdot f(b)$
- $n = \sum_{d|n} \varphi(d)$ 。由莫比乌斯反演可得。
- 若 $n = p^k$ ，其中 p 是质数，那么 $\varphi(n) = p^k - p^{k-1}$ 。
- 由唯一分解定理，设 $n = \prod_{i=1}^n p_i^{k_i}$ ，其中 p_i 是质数，有

$$\varphi(n) = n \cdot \prod_{i=1}^s \frac{p_i - 1}{p_i}。$$

欧拉函数

求解

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

■ 求解单个欧拉函数值：直接质因数分解就好了

```
1  int phi(int n) {  
2      int ans = n;  
3      for (int i = 2; i * i <= n; i++)  
4          if (n % i == 0) {  
5              ans = ans / i * (i - 1);  
6              while (n % i == 0) n /= i;  
7          }  
8      if (n > 1) ans = ans / n * (n - 1);  
9      return ans;  
10 }
```

欧拉函数

求解

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

演

数论分块

结束

■ 求解 $1 \sim n$ 的欧拉函数值：线性筛

```

1 void init() {
2     int cnt = 0;
3     phi[1] = 1;
4     for (int i = 2; i < MAXN; i++) {
5         if (!inp[i]) {
6             prime[++cnt] = i;
7             phi[i] = i - 1;
8         }
9
10        for (int j = 1; j <= cnt && i * prime[j] < MAXN; j++) {
11            inp[i * prime[j]] = true;
12            if (i % prime[j] == 0) {
13                phi[i * prime[j]] = phi[i] * prime[j];
14                break;
15            }
16            else
17                phi[i * prime[j]] = phi[i] * (prime[j] - 1);
18        }
19    }
20 }

```

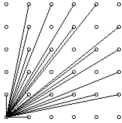
欧拉函数

例题

题目链接: P2158 [SDOI2008] 仪仗队 - 洛谷

题目描述

作为体育委员，C君负责这次运动会仪仗队的训练。仪仗队是由学生组成的 $N * N$ 的方阵，为了保证队伍在行进中整齐划一，C君会跟在仪仗队的左后方，根据其视线所及的学生人数来判断队伍是否整齐(如下图)。



现在，C君希望你告诉他队伍整齐时能看到的学生人数。

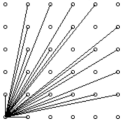
欧拉函数

例题

题目链接: P2158 [SDOI2008] 仪仗队 - 洛谷

题目描述

作为体育委员, C君负责这次运动会仪仗队的训练。仪仗队是由学生组成的 $N * N$ 的方阵, 为了保证队伍在行进中整齐划一, C君会跟在仪仗队的左后方, 根据其视线所及的学生人数来判断队伍是否整齐(如下图)。



现在, C君希望你告诉他队伍整齐时能看到的学生人数。

题解

- 什么时候一个人会被另一个人挡住？

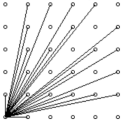
欧拉函数

例题

题目链接: P2158 [SDOI2008] 仪仗队 - 洛谷

题目描述

作为体育委员，C君负责这次运动会仪仗队的训练。仪仗队是由学生组成的 $N * N$ 的方阵，为了保证队伍在行进中整齐划一，C君会跟在仪仗队的左后方，根据其视线所及的学生人数来判断队伍是否整齐(如下图)。



现在，C君希望你告诉他队伍整齐时能看到的学生人数。

题解

- 什么时候一个人会被另一个人挡住？
- 假设俩人坐标为 (x_1, y_1) , (x_2, y_2) , $x_1 \leq x_2$
- 显然当且仅当 $\frac{y_1}{x_1} = \frac{y_2}{x_2}$ 时 2 会被 1 挡住

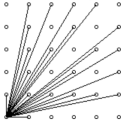
欧拉函数

例题

题目链接: P2158 [SDOI2008] 仪仗队 - 洛谷

题目描述

作为体育委员, C君负责这次运动会仪仗队的训练。仪仗队是由学生组成的 $N * N$ 的方阵, 为了保证队伍在行进中整齐划一, C君会跟在仪仗队的左后方, 根据其视线所及的学生人数来判断队伍是否整齐(如下图)。



现在, C君希望你告诉他队伍整齐时能看到的学生人数。

题解

- 什么时候一个人会被另一个人挡住?
- 假设俩人坐标为 (x_1, y_1) , (x_2, y_2) , $x_1 \leq x_2$
- 显然当且仅当 $\frac{y_1}{x_1} = \frac{y_2}{x_2}$ 时 2 会被 1 挡住
- 只考虑 $y < x$ 的部分, 可得第 i 列只有 $\phi(i)$ 个人没被挡住, 即可被看见

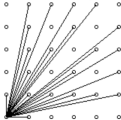
欧拉函数

例题

题目链接: P2158 [SDOI2008] 仪仗队 - 洛谷

题目描述

作为体育委员, C君负责这次运动会仪仗队的训练。仪仗队是由学生组成的 $N * N$ 的方阵, 为了保证队伍在行进中整齐划一, C君会跟在仪仗队的左后方, 根据其视线所及的学生人数来判断队伍是否整齐(如下图)。



现在, C君希望你告诉他队伍整齐时能看到的学生人数。

题解

- 什么时候一个人会被另一个人挡住?
- 假设俩人坐标为 (x_1, y_1) , (x_2, y_2) , $x_1 \leq x_2$
- 显然当且仅当 $\frac{y_1}{x_1} = \frac{y_2}{x_2}$ 时 2 会被 1 挡住
- 只考虑 $y < x$ 的部分, 可得第 i 列只有 $\varphi(i)$ 个人没被挡住, 即可被看见
- 所以 $ans = 2 * (\sum_{i=1}^n \varphi(i)) + 1$

费马小定理

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

演

数论分块

结束

定理（费马小定理）

若 p 为素数， $\gcd(a, p) = 1$ ，则 $a^{p-1} \equiv 1 \pmod{p}$ 。

另一个形式：对于任意整数 a ，有 $a^p \equiv a \pmod{p}$ 。

费马小定理

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

演

数论分块

结束

定理（费马小定理）

若 p 为素数, $\gcd(a, p) = 1$, 则 $a^{p-1} \equiv 1 \pmod{p}$ 。

另一个形式: 对于任意整数 a , 有 $a^p \equiv a \pmod{p}$ 。

费马小定理常用于结合快速幂求解乘法逆元:

$$a^{p-2} \cdot a \equiv 1 \pmod{p}, (\gcd(a, p) = 1)$$

费马小定理

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

演

数论分块

结束

定理（费马小定理）

若 p 为素数, $\gcd(a, p) = 1$, 则 $a^{p-1} \equiv 1 \pmod{p}$ 。

另一个形式: 对于任意整数 a , 有 $a^p \equiv a \pmod{p}$ 。

费马小定理常用于结合快速幂求解乘法逆元:

$$a^{p-2} \cdot a \equiv 1 \pmod{p}, (\gcd(a, p) = 1)$$

证明:

我有一个对这个命题的十分美妙的证明, 这里空白太小, 我写不下了。

费马小定理

证明

Proof.

设一个质数为 p , 且 $\gcd(a, p) = 1$ 。

构造一个序列: $A = \{1, 2, 3 \dots, p-1\}$, 这个序列有着这样一个性质:

$$\prod_{i=1}^n A_i \equiv \prod_{i=1}^n (A_i \times a) \pmod{p}$$

费马小定理

证明

Proof.

设一个质数为 p , 且 $\gcd(a, p) = 1$ 。

构造一个序列: $A = \{1, 2, 3 \dots, p-1\}$, 这个序列有着这样一个性质:

$$\prod_{i=1}^n A_i \equiv \prod_{i=1}^n (A_i \times a) \pmod{p}$$

思路: 反证法可证得 $\forall i \neq j, A_i a \not\equiv A_j a \pmod{p}$

那么 $\{A_i a\}$ 将取到 $1 \sim p-1$ 内的所有值。

费马小定理

证明

Proof.

设一个质数为 p , 且 $\gcd(a, p) = 1$ 。

构造一个序列: $A = \{1, 2, 3 \dots, p-1\}$, 这个序列有着这样一个性质:

$$\prod_{i=1}^n A_i \equiv \prod_{i=1}^n (A_i \times a) \pmod{p}$$

思路: 反证法可证得 $\forall i \neq j, A_i a \not\equiv A_j a \pmod{p}$

那么 $\{A_i a\}$ 将取到 $1 \sim p-1$ 内的所有值。

$$\begin{aligned} \prod_{i=1}^n (A_i \times a) &\equiv \prod_{i=1}^n A_i \pmod{p} \\ \Rightarrow a^{p-1} \prod_{i=1}^n A_i &\equiv \prod_{i=1}^n A_i \pmod{p} \\ \Rightarrow a^{p-1} &\equiv 1 \pmod{p} \end{aligned}$$



欧拉定理

定理（欧拉定理）

若 $\gcd(a, m) = 1$, 则 $a^{\varphi(m)} \equiv 1 \pmod{m}$ 。

则当 m 为质数时, $\varphi(m) = m - 1$, 则可得 $a^{m-1} \equiv 1 \pmod{m}$, 即得到了费马小定理。

欧拉定理

定理（欧拉定理）

若 $\gcd(a, m) = 1$ ，则 $a^{\varphi(m)} \equiv 1 \pmod{m}$ 。

则当 m 为质数时， $\varphi(m) = m - 1$ ，则可得 $a^{m-1} \equiv 1 \pmod{m}$ ，即得到了费马小定理。

证明： 欧拉定理与费马小定理及证明

拓展欧拉定理

欧拉定理只能在 $\gcd(a, p) = 1$ 时使用，限制很大。

拓展欧拉定理

欧拉定理只能在 $\gcd(a, p) = 1$ 时使用，限制很大。

定理（拓展欧拉定理）

$$a^b \equiv \begin{cases} a^{b \bmod \varphi(p)}, & \gcd(a, p) = 1 \\ a^b, & \gcd(a, p) \neq 1, b < \varphi(p) \\ a^{b \bmod \varphi(p) + \varphi(p)}, & \gcd(a, p) \neq 1, b \geq \varphi(p) \end{cases} \pmod{p}$$

拓展欧拉定理

欧拉定理只能在 $\gcd(a, p) = 1$ 时使用，限制很大。

定理（拓展欧拉定理）

$$a^b \equiv \begin{cases} a^{b \bmod \varphi(p)}, & \gcd(a, p) = 1 \\ a^b, & \gcd(a, p) \neq 1, b < \varphi(p) \\ a^{b \bmod \varphi(p) + \varphi(p)}, & \gcd(a, p) \neq 1, b \geq \varphi(p) \end{cases} \pmod{p}$$

完美满足所有情况，非常好用。—(虽然我没怎么用过)—

拓展欧拉定理

欧拉定理只能在 $\gcd(a, p) = 1$ 时使用，限制很大。

定理（拓展欧拉定理）

$$a^b \equiv \begin{cases} a^{b \bmod \varphi(p)}, & \gcd(a, p) = 1 \\ a^b, & \gcd(a, p) \neq 1, b < \varphi(p) \\ a^{b \bmod \varphi(p) + \varphi(p)}, & \gcd(a, p) \neq 1, b \geq \varphi(p) \end{cases} \pmod{p}$$

完美满足所有情况，非常好用。—(虽然我没怎么用过)—

证明：太长了我写不下 我不会

拓展欧拉定理

欧拉定理只能在 $\gcd(a, p) = 1$ 时使用，限制很大。

定理（拓展欧拉定理）

$$a^b \equiv \begin{cases} a^{b \bmod \varphi(p)}, & \gcd(a, p) = 1 \\ a^b, & \gcd(a, p) \neq 1, b < \varphi(p) \\ a^{b \bmod \varphi(p) + \varphi(p)}, & \gcd(a, p) \neq 1, b \geq \varphi(p) \end{cases} \pmod{p}$$

完美满足所有情况，非常好用。—(虽然我没怎么用过)—

证明：太长了我写不下 我不会

常用于欧拉降幂。

拓展欧拉定理

例题

题目链接: [P1066 AB%P - XDOJ](#)

题目描述

求解:

$$(\dots((A^{B_0})^{B_1})\dots)^{B_{n-1}} \bmod P$$

其中, $B_i = B_{i-1}^2 - 1 (i > 0)$, $P = 1e9 + 7$ 。

数据范围: $0 < A < 2^{31}, 0 \leq n \leq 1e5, 1 < B_0 < 2^{31}$

拓展欧拉定理

例题

题目链接: [P1066 AB%P - XDOJ](#)

题目描述

求解:

$$(\dots((A^{B_0})^{B_1})\dots)^{B_{n-1}} \bmod P$$

其中, $B_i = B_{i-1}^2 - 1 (i > 0)$, $P = 1e9 + 7$ 。

数据范围: $0 < A < 2^{31}, 0 \leq n \leq 1e5, 1 < B_0 < 2^{31}$

题解

- 即求 $A^{\sum_{i=0}^{n-1} B_i} \bmod p$, 但是 B_i 是指数级增加, 无法直接求出

拓展欧拉定理

例题

题目链接: [P1066 AB%P - XDOJ](#)

题目描述

求解:

$$(\dots((A^{B_0})^{B_1})\dots)^{B_{n-1}} \bmod P$$

其中, $B_i = B_{i-1}^2 - 1 (i > 0)$, $P = 1e9 + 7$ 。

数据范围: $0 < A < 2^{31}, 0 \leq n \leq 1e5, 1 < B_0 < 2^{31}$

题解

- 即求 $A^{\sum_{i=0}^{n-1} B_i} \bmod p$, 但是 B_i 是指数级增加, 无法直接求出
- 显然欧拉降幂, 考虑适用条件
- $P = 1e9 + 7$ 是质数, 所以要么 $\gcd(A, P) = 1$, 要么 $P|A$

拓展欧拉定理

例题

题目链接: P1066 AB%P - XDOJ

题目描述

求解:

$$(\dots((A^{B_0})^{B_1})\dots)^{B_{n-1}} \bmod P$$

其中, $B_i = B_{i-1}^2 - 1 (i > 0)$, $P = 1e9 + 7$ 。

数据范围: $0 < A < 2^{31}, 0 \leq n \leq 1e5, 1 < B_0 < 2^{31}$

题解

- 即求 $A^{\sum_{i=0}^{n-1} B_i} \bmod p$, 但是 B_i 是指数级增加, 无法直接求出
- 显然欧拉降幂, 考虑适用条件
- $P = 1e9 + 7$ 是质数, 所以要么 $\gcd(A, P) = 1$, 要么 $P | A$
- 显然二者条件下, $A^x \equiv A^{x \bmod \varphi(P)} \pmod{P}$ 均成立
- 所以先求出 $\sum_{i=0}^{n-1} B_i \bmod \varphi(P)$ ($\varphi(P) = P - 1$) 即可。

乘法逆元

简介

乘法逆元：如果一个线性同余方程 $ax \equiv 1 \pmod{b}$ ，则 x 称为 $a \bmod b$ 的逆元，记作 a^{-1} 。

用处：

- 求解 $\frac{A}{B} \bmod P$
- 避免浮点运算，提高精度
- 某些题目的要求

乘法逆元

简介

乘法逆元：如果一个线性同余方程 $ax \equiv 1 \pmod{b}$ ，则 x 称为 $a \bmod b$ 的逆元，记作 a^{-1} 。

用处：

- 求解 $\frac{A}{B} \bmod P$
- 避免浮点运算，提高精度
- 某些题目的要求

求解：

- 快速幂方法
- 拓展欧几里得方法

乘法逆元

简介

乘法逆元：如果一个线性同余方程 $ax \equiv 1 \pmod{b}$ ，则 x 称为 $a \bmod b$ 的逆元，记作 a^{-1} 。

用处：

- 求解 $\frac{A}{B} \bmod P$
- 避免浮点运算，提高精度
- 某些题目的要求

求解：

- 快速幂方法
- 拓展欧几里得方法
- 依据个人喜好

乘法逆元

求解单个

快速幂

■ **事实：**

$$\begin{aligned} a^{b-1} &\equiv 1 \pmod{p} \\ \Rightarrow a \cdot a^{b-2} &\equiv 1 \pmod{p} \\ \Rightarrow x &\equiv a^{b-2} \pmod{p} \end{aligned}$$

乘法逆元

求解单个

快速幂

■ 事实：

$$\begin{aligned} a^{b-1} &\equiv 1 \pmod{p} \\ \Rightarrow a \cdot a^{b-2} &\equiv 1 \pmod{p} \\ \Rightarrow x &\equiv a^{b-2} \pmod{p} \end{aligned}$$

■ 算法实现：

```

1 inline int qpow(long long a, int b) {
2     int ans = 1;
3     a = (a % p + p) % p;
4     for (; b; b >>= 1) {
5         if (b & 1) ans = (a * ans) % p;
6         a = (a * a) % p;
7     }
8     return ans;
9 }
10 // int inv = qpow(a, p - 2);

```

乘法逆元

求解单个

快速幂

■ 事实：

$$\begin{aligned} a^{b-1} &\equiv 1 \pmod{p} \\ \Rightarrow a \cdot a^{b-2} &\equiv 1 \pmod{p} \\ \Rightarrow x &\equiv a^{b-2} \pmod{p} \end{aligned}$$

■ 算法实现：

```

1 inline int qpow(long long a, int b) {
2     int ans = 1;
3     a = (a % p + p) % p;
4     for (; b >= 1) {
5         if (b & 1) ans = (a * ans) % p;
6         a = (a * a) % p;
7     }
8     return ans;
9 }
10 // int inv = qpow(a, p - 2);

```

■ 时间复杂度： $O(\log n)$

乘法逆元

求解单个

拓展欧几里得：

■ 事实：

$$ax \equiv 1 \pmod{p}$$

$$ax = (-y) \cdot p + 1$$

求解： $ax + py = 1$

乘法逆元

求解单个

拓展欧几里得：

■ 事实：

$$ax \equiv 1 \pmod{p}$$

$$ax = (-y) \cdot p + 1$$

$$\text{求解： } ax + py = 1$$

■ 算法实现：

```

1  int exgcd(int a, int b, int& x, int& y) {
2      if (b == 0) {
3          x = 1, y = 0;
4          return a;
5      }
6      int g = exgcd(b, a % b, y, x);
7      y -= a / b * x;
8      return g;
9  }
10 // exgcd(a, p, x, y);
11 // int inv = (x % p + p) % p;
```

乘法逆元

求解单个

拓展欧几里得：

■ 事实：

$$ax \equiv 1 \pmod{p}$$

$$ax = (-y) \cdot p + 1$$

$$\text{求解: } ax + py = 1$$

■ 算法实现：

```
1 int exgcd(int a, int b, int& x, int& y) {  
2     if (b == 0) {  
3         x = 1, y = 0;  
4         return a;  
5     }  
6     int g = exgcd(b, a % b, y, x);  
7     y -= a / b * x;  
8     return g;  
9 }  
10 // exgcd(a, p, x, y);  
11 // int inv = (x % p + p) % p;
```

■ 时间复杂度： $O(\log n)$

乘法逆元

求解 $1 \sim n$

线性时间复杂度求出 $1, 2, \dots, n$ 中每个数关于 p 的逆元。

乘法逆元

求解 $1 \sim n$

线性时间复杂度求出 $1, 2, \dots, n$ 中每个数关于 p 的逆元。

Proof.

事实: $\forall p \in \mathbb{Z}, 1 \times 1 \equiv 1 \pmod{p}$ 恒成立, 即 \pmod{p} 意义下 $1^{-1} = 1$ 。

乘法逆元

求解 $1 \sim n$

线性时间复杂度求出 $1, 2, \dots, n$ 中每个数关于 p 的逆元。

Proof.

事实: $\forall p \in \mathbb{Z}, 1 \times 1 \equiv 1 \pmod{p}$ 恒成立, 即 \pmod{p} 意义下 $1^{-1} = 1$ 。

思路: 通过 $1 \sim i-1$ 的逆元在 $O(1)$ 的复杂度下求解 i^{-1} 。

乘法逆元

求解 $1 \sim n$

线性时间复杂度求出 $1, 2, \dots, n$ 中每个数关于 p 的逆元。

Proof.

事实: $\forall p \in \mathbb{Z}, 1 \times 1 \equiv 1 \pmod{p}$ 恒成立, 即 \pmod{p} 意义下 $1^{-1} = 1$ 。

思路: 通过 $1 \sim i-1$ 的逆元在 $O(1)$ 的复杂度下求解 i^{-1} 。

设:

$$k = \left\lfloor \frac{p}{i} \right\rfloor, j = p \bmod i, \text{ 则 } p = ki + j$$

乘法逆元

求解 $1 \sim n$ 线性时间复杂度求出 $1, 2, \dots, n$ 中每个数关于 p 的逆元。

Proof.

事实: $\forall p \in \mathbb{Z}, 1 \times 1 \equiv 1 \pmod{p}$ 恒成立, 即 \pmod{p} 意义下 $1^{-1} = 1$ 。**思路:** 通过 $1 \sim i-1$ 的逆元在 $O(1)$ 的复杂度下求解 i^{-1} 。

设:

$$k = \left\lfloor \frac{p}{i} \right\rfloor, j = p \bmod i, \text{ 则 } p = ki + j$$

我们有:

$$p = ki + j \equiv 0 \pmod{p}$$

乘法逆元

求解 $1 \sim n$ 线性时间复杂度求出 $1, 2, \dots, n$ 中每个数关于 p 的逆元。

Proof.

事实: $\forall p \in \mathbb{Z}, 1 \times 1 \equiv 1 \pmod{p}$ 恒成立, 即 \pmod{p} 意义下 $1^{-1} = 1$ 。**思路:** 通过 $1 \sim i-1$ 的逆元在 $O(1)$ 的复杂度下求解 i^{-1} 。

设:

$$k = \lfloor \frac{p}{i} \rfloor, j = p \bmod i, \text{ 则 } p = ki + j$$

我们有:

$$p = ki + j \equiv 0 \pmod{p}$$

$$ki \cdot i^{-1} j^{-1} + j \cdot i^{-1} j^{-1} \equiv 0 \cdot i^{-1} j^{-1} \pmod{p}$$

$$k \cdot j^{-1} + i^{-1} \equiv 0 \pmod{p}$$

乘法逆元

求解 $1 \sim n$ 线性时间复杂度求出 $1, 2, \dots, n$ 中每个数关于 p 的逆元。

Proof.

事实: $\forall p \in \mathbb{Z}, 1 \times 1 \equiv 1 \pmod{p}$ 恒成立, 即 \pmod{p} 意义下 $1^{-1} = 1$ 。**思路:** 通过 $1 \sim i-1$ 的逆元在 $O(1)$ 的复杂度下求解 i^{-1} 。

设:

$$k = \lfloor \frac{p}{i} \rfloor, j = p \bmod i, \text{ 则 } p = ki + j$$

我们有:

$$p = ki + j \equiv 0 \pmod{p}$$

$$ki \cdot i^{-1} j^{-1} + j \cdot i^{-1} j^{-1} \equiv 0 \cdot i^{-1} j^{-1} \pmod{p}$$

$$k \cdot j^{-1} + i^{-1} \equiv 0 \pmod{p}$$

$$i^{-1} \equiv -k \cdot j^{-1} \pmod{p}$$

$$i^{-1} \equiv -k \cdot \lfloor \frac{p}{i} \rfloor \pmod{p}$$



1 概述

- 简介
- 符号约定

2 整除与同余

- 整除
- 同余
- 快速幂

3 素数

- 素数
- 筛法

4 欧几里得

- 因数
- 最大公约数

- 最小公倍数
- 拓展欧几里得

5 欧拉函数

- 欧拉函数
- 费马小定理
- 欧拉定理

6 乘法逆元

- 简介
- 单个求解
- 多个求解

7 莫比乌斯反演

- ## ■ 数论分块

8 结束

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

结束

引理

$$\forall a, b, c \in \mathbb{Z}, \left\lfloor \frac{a}{bc} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{a}{b} \right\rfloor}{c} \right\rfloor$$

数论分块

引理

$$\forall a, b, c \in \mathbb{Z}, \left\lfloor \frac{a}{bc} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{a}{b} \right\rfloor}{c} \right\rfloor$$

Proof.

$$\begin{aligned} \frac{a}{b} &= \left\lfloor \frac{a}{b} \right\rfloor + r (0 \leq r < 1) \\ \Rightarrow \left\lfloor \frac{a}{bc} \right\rfloor &= \left\lfloor \frac{a}{b} \cdot \frac{1}{c} \right\rfloor = \left\lfloor \frac{1}{c} \left(\left\lfloor \frac{a}{b} \right\rfloor + r \right) \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{a}{b} \right\rfloor}{c} + \frac{r}{c} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{a}{b} \right\rfloor}{c} \right\rfloor \end{aligned}$$



数论分块

引理

$$\forall a, b, c \in \mathbb{Z}, \left\lfloor \frac{a}{bc} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{a}{b} \right\rfloor}{c} \right\rfloor$$

Proof.

$$\frac{a}{b} = \left\lfloor \frac{a}{b} \right\rfloor + r (0 \leq r < 1)$$

$$\Rightarrow \left\lfloor \frac{a}{bc} \right\rfloor = \left\lfloor \frac{a}{b} \cdot \frac{1}{c} \right\rfloor = \left\lfloor \frac{1}{c} \left(\left\lfloor \frac{a}{b} \right\rfloor + r \right) \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{a}{b} \right\rfloor}{c} + \frac{r}{c} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{a}{b} \right\rfloor}{c} \right\rfloor$$



引理

$$\forall n \in \mathbb{N}_+, \left| \left\{ \left\lfloor \frac{n}{d} \right\rfloor \mid d \in \mathbb{N}_+, d \leq n \right\} \right| \leq \lfloor 2\sqrt{n} \rfloor$$

$|V|$ 表示集合 V 的元素个数。

数论分块

简介

概述

简介

符号约定

整除与同余

整除

同余

快速幂

素数

素数

筛法

欧几里得

因数

最大公约数

最小公倍数

拓展欧几里得

欧拉函数

欧拉函数

费马小定理

欧拉定理

乘法逆元

简介

单个求解

多个求解

莫比乌斯反演

数论分块

数论分块

结束

求解类似于 $\sum_{i=1}^n f(\lfloor \frac{n}{i} \rfloor)$ 的式子。(假设 $f(x)$ 可以 $O(1)$ 求出)

- 直接求和，时间复杂度为 $O(n)$ ，可通过数据约为 $n \leq 1e8$ 。
- 事实上， $\lfloor \frac{n}{i} \rfloor$ 只有 $O(\sqrt{n})$ 种，能否简化求和次数？

求解类似于 $\sum_{i=1}^n f(\lfloor \frac{n}{i} \rfloor)$ 的式子。(假设 $f(x)$ 可以 $O(1)$ 求出)

- 直接求和，时间复杂度为 $O(n)$ ，可通过数据约为 $n \leq 1e8$ 。
- 事实上， $\lfloor \frac{n}{i} \rfloor$ 只有 $O(\sqrt{n})$ 种，能否简化求和次数？

对于任意一个 $i (i \leq n)$ ，我们需要找到一个最大的 $j (i \leq j \leq n)$ ，使得：

$$\left\lfloor \frac{n}{i} \right\rfloor = \left\lfloor \frac{n}{j} \right\rfloor$$

可得：

$$j = \left\lfloor \frac{n}{\left\lfloor \frac{n}{i} \right\rfloor} \right\rfloor$$

数论分块

证明

Proof.

$$\begin{aligned}\left\lfloor \frac{n}{i} \right\rfloor &\leq \frac{n}{i} \\ \Rightarrow \left\lfloor \frac{n}{\left\lfloor \frac{n}{i} \right\rfloor} \right\rfloor &\geq \left\lfloor \frac{n}{\frac{n}{i}} \right\rfloor = \lfloor i \rfloor = i \\ \Rightarrow i &\leq \left\lfloor \frac{n}{\left\lfloor \frac{n}{i} \right\rfloor} \right\rfloor = j\end{aligned}$$



数论分块

证明

Proof.

$$\left\lfloor \frac{n}{i} \right\rfloor \leq \frac{n}{i}$$

$$\Rightarrow \left\lfloor \frac{n}{\left\lfloor \frac{n}{i} \right\rfloor} \right\rfloor \geq \left\lfloor \frac{n}{\frac{n}{i}} \right\rfloor = \lfloor i \rfloor = i$$

$$\Rightarrow i \leq \left\lfloor \frac{n}{\left\lfloor \frac{n}{i} \right\rfloor} \right\rfloor = j$$



■ 基本使用：

```

1  for (int l = 1, r; l <= n; l = r + 1) {
2      int t = n / l;
3      r = n / t;
4      // ... 若 l <= i <= r, 那么 n / i == t 为真
5  }
```

数论分块

例题

题目链接: [P2261 \[CQOI2007\] 余数求和 - 洛谷](#)

题目描述

给定正整数 n 和 k , 求解:

$$G(n, k) = \sum_{i=1}^n k \bmod i$$

数据范围: $1 \leq n, k \leq 10^9$

数论分块

例题

题目链接: [P2261 \[CQOI2007\] 余数求和 - 洛谷](#)

题目描述

给定正整数 n 和 k , 求解:

$$G(n, k) = \sum_{i=1}^n k \bmod i$$

数据范围: $1 \leq n, k \leq 10^9$

题解

- 直接求和, $O(n)$ 时间复杂度会 TLE。

数论分块

例题

题目链接: [P2261 \[CQOI2007\] 余数求和 - 洛谷](#)

题目描述

给定正整数 n 和 k , 求解:

$$G(n, k) = \sum_{i=1}^n k \bmod i$$

数据范围: $1 \leq n, k \leq 10^9$

题解

- 直接求和, $O(n)$ 时间复杂度会 TLE。
- 转化为整除形式, 然后数论分块:

$$\text{ans} = \sum_{i=1}^n (k \bmod i) = \sum_{i=1}^n k - i \left\lfloor \frac{k}{i} \right\rfloor = n \cdot k - \sum_{i=1}^{\min(n, k)} i \left\lfloor \frac{k}{i} \right\rfloor$$

数论分块

例题

■ 算法实现：

```
1  #include<iostream>
2  using namespace std;
3
4  inline long long sum(long long i, long long j){
5      return (j * (j + 1) / 2 - i * (i - 1) / 2);
6  }
7
8  int main(){
9      long long n, k, ans;
10     cin >> n >> k;
11     ans = n * k;
12     for (long long i = 1, j; i <= k && i <= n; i = j + 1){
13         long long t = k / i;
14         j = min(k / t, n);
15         ans -= t * sum(i, j);
16     }
17     cout << ans ;
18     return 0;
19 }
```

数论分块

例题

题目推荐：

- P2261 [CQOI2007] 余数求和 - 洛谷
- P2260 [清华集训 2012] 模积和 - 洛谷
- Debugging - Codeforces

1 概述

- 简介
- 符号约定

2 整除与同余

- 整除
- 同余
- 快速幂

3 素数

- 素数
- 筛法

4 欧几里得

- 因数
- 最大公约数

- 最小公倍数
- 拓展欧几里得

5 欧拉函数

- 欧拉函数
- 费马小定理
- 欧拉定理

6 乘法逆元

- 简介
- 单个求解
- 多个求解

7 莫比乌斯反演

- ## ■ 数论分块

8 结束

