

类欧几里得算
法

洪华敦

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

类欧几里得算法

洪华敦

WC2016

需要用的的一些数学符号

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

需要用的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- $[a]$ 若表达式 a 为真，则 $[a] = 1$ ，否则 $[a] = 0$

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- $[a]$ 若表达式 a 为真，则 $[a] = 1$ ，否则 $[a] = 0$
- 例如 $[1 + 1 = 2] = 1$

需要用的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- $[a]$ 若表达式 a 为真，则 $[a] = 1$ ，否则 $[a] = 0$
- 例如 $[1 + 1 = 2] = 1$
- % 取模

需要用的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- $[a]$ 若表达式 a 为真，则 $[a] = 1$ ，否则 $[a] = 0$
- 例如 $[1 + 1 = 2] = 1$
- % 取模
- $S_d(n) = \sum_{x=1}^n x^d$

需要用的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- $[a]$ 若表达式 a 为真，则 $[a] = 1$ ，否则 $[a] = 0$
- 例如 $[1 + 1 = 2] = 1$
- % 取模
- $S_d(n) = \sum_{x=1}^n x^d$
- $\lfloor \frac{a}{b} \rfloor$ $\frac{a}{b}$ 下取整

欧几里得算法

类欧几里得算
法

洪华敦

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

洪华敦

需要用到的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 对于 $b \neq 0$, 有 $\gcd(a, b) = \gcd(b, a \% b)$

复杂度证明

类欧几里得算
法

洪华敦

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 为了更好地讲后面的东西，我们来证明下欧几里得算法的复杂度

- 为了更好地讲后面的东西，我们来证明下欧几里得算法的复杂度
- 考虑最坏情况， $a \leq 2 * b - 1$

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 为了更好地讲后面的东西，我们来证明下欧几里得算法的复杂度
- 考虑最坏情况， $a \leq 2 * b - 1$
- 那么有 $\gcd(a, b) = \gcd(b, a - b)$

需要用到的一些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 为了更好地讲后面的东西，我们来证明下欧几里得算法的复杂度
- 考虑最坏情况， $a \leq 2 * b - 1$
- 那么有 $\gcd(a, b) = \gcd(b, a - b)$
- 将 a, b 设为比较接近的斐波那契数，那么 $a = F_n, b = F_{n-1}$ ，则
有 $\gcd(F_n, F_{n-1}) = \gcd(F_{n-1}, F_{n-2})$

需要用到的一些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 为了更好地讲后面的东西，我们来证明下欧几里得算法的复杂度
- 考虑最坏情况， $a \leq 2 * b - 1$
- 那么有 $\gcd(a, b) = \gcd(b, a - b)$
- 将 a, b 设为比较接近的斐波那契数，那么 $a = F_n, b = F_{n-1}$ ，则
有 $\gcd(F_n, F_{n-1}) = \gcd(F_{n-1}, F_{n-2})$
- 显然 n 是 $O(\log(F_n))$ 的

需要用到的一些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 为了更好地讲后面的东西，我们来证明下欧几里得算法的复杂度
- 考虑最坏情况， $a \leq 2 * b - 1$
- 那么有 $\gcd(a, b) = \gcd(b, a - b)$
- 将 a, b 设为比较接近的斐波那契数，那么 $a = F_n, b = F_{n-1}$ ，则
有 $\gcd(F_n, F_{n-1}) = \gcd(F_{n-1}, F_{n-2})$
- 显然 n 是 $O(\log(F_n))$ 的
- 所以这个算法的复杂度是 $O(\log(a))$ 的

一个问题

洪华敦

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

洪华敦

需要用的一些
数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 我们来看这样一个问题

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 我们来看这样一个问题
- $\sum_{x=0}^n \left\lfloor \frac{ax+b}{c} \right\rfloor$

做法

类欧几里得算法

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 首先，如果 $a \geq c$ ，设 $d = \lfloor \frac{a}{c} \rfloor$ 则有：

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩
展

- 首先，如果 $a \geq c$ ，设 $d = \lfloor \frac{a}{c} \rfloor$ 则有：
- $\sum_{x=0}^n \left\lfloor \frac{ax+b}{c} \right\rfloor = \sum_{x=0}^n \left\lfloor \frac{(a \% c)x + b}{c} \right\rfloor + d * x$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩
展

- 首先，如果 $a >= c$, 设 $d = \lfloor \frac{a}{c} \rfloor$ 则有:
- $\sum_{x=0}^n \left\lfloor \frac{ax+b}{c} \right\rfloor = \sum_{x=0}^n \left\lfloor \frac{(a \% c)x + b}{c} \right\rfloor + d * x$
- $= d * S_1(n) + \sum_{x=0}^n \left\lfloor \frac{(a \% c)x + b}{c} \right\rfloor$

做法

类欧几里得算法

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 同理，如果有 $b \geq c$ ，设 $d = \lfloor \frac{b}{c} \rfloor$ 则有：

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 同理，如果有 $b \geq c$ ，设 $d = \lfloor \frac{b}{c} \rfloor$ 则有：
- $\sum_{x=0}^n \left\lfloor \frac{ax+b}{c} \right\rfloor = \sum_{x=0}^n \left\lfloor \frac{ax+(b \% c)}{c} \right\rfloor + d$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩
展

- 同理，如果有 $b \geq c$ ，设 $d = \lfloor \frac{b}{c} \rfloor$ 则有：
- $\sum_{x=0}^n \left\lfloor \frac{ax+b}{c} \right\rfloor = \sum_{x=0}^n \left\lfloor \frac{ax+(b \% c)}{c} \right\rfloor + d$
- $= d * S_0(n) + \sum_{x=0}^n \left\lfloor \frac{ax+(b \% c)}{c} \right\rfloor$

做法

类欧几里得算法

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 经过上面两个计算后， 我们有 $a < c, b < c$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 经过上面两个计算后，我们有 $a < c, b < c$
- 于是我们的问题规模从 (a, c, b, n) 变成了 $(a \% c, c, b \% c, n)$

做法

类欧几里得算法

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用的一些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 令 $M = \left\lfloor \frac{an+b}{c} \right\rfloor$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算
法一些简单的扩
展

- 令 $M = \left\lfloor \frac{an+b}{c} \right\rfloor$
- 原式 = $\sum_{x=0}^n \sum_{y=0}^M [y < \left\lfloor \frac{ax+b}{c} \right\rfloor]$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩
展

- 令 $M = \left\lfloor \frac{an+b}{c} \right\rfloor$
- 原式 = $\sum_{x=0}^n \sum_{y=0}^M [y < \left\lfloor \frac{ax+b}{c} \right\rfloor]$
- 我们来化简下这个式子 $y < \left\lfloor \frac{ax+b}{c} \right\rfloor$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 令 $M = \left\lfloor \frac{an+b}{c} \right\rfloor$
- 原式 = $\sum_{x=0}^n \sum_{y=0}^M [y < \left\lfloor \frac{ax+b}{c} \right\rfloor]$
- 我们来化简下这个式子 $y < \left\lfloor \frac{ax+b}{c} \right\rfloor$
- $c * (y + 1) \leq a * x + b$

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 令 $M = \left\lfloor \frac{an+b}{c} \right\rfloor$
- 原式 = $\sum_{x=0}^n \sum_{y=0}^M [y < \left\lfloor \frac{ax+b}{c} \right\rfloor]$
- 我们来化简下这个式子 $y < \left\lfloor \frac{ax+b}{c} \right\rfloor$
- $c * (y + 1) \leq a * x + b$
- $c * y + c - b \leq a * x$

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 令 $M = \left\lfloor \frac{an+b}{c} \right\rfloor$
- 原式 = $\sum_{x=0}^n \sum_{y=0}^M [y < \left\lfloor \frac{ax+b}{c} \right\rfloor]$
- 我们来化简下这个式子 $y < \left\lfloor \frac{ax+b}{c} \right\rfloor$
- $c * (y + 1) \leq a * x + b$
- $c * y + c - b \leq a * x$
- $c * y + c - b - 1 < a * x$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算
法一些简单的扩
展

- 令 $M = \left\lfloor \frac{an+b}{c} \right\rfloor$
- 原式 = $\sum_{x=0}^n \sum_{y=0}^M [y < \left\lfloor \frac{ax+b}{c} \right\rfloor]$
- 我们来化简下这个式子 $y < \left\lfloor \frac{ax+b}{c} \right\rfloor$
- $c * (y + 1) \leq a * x + b$
- $c * y + c - b \leq a * x$
- $c * y + c - b - 1 < a * x$
- $x > \left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor$

做法

类欧几里得算法

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算
法一些简单的扩
展

- 所以原式= $\sum_{y=0}^M \sum_{x=0}^n [x > \left\lfloor \frac{c*y + c - b - 1}{a} \right\rfloor]$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩
展

- 所以原式 = $\sum_{y=0}^M \sum_{x=0}^n [x > \left\lfloor \frac{c*y + c - b - 1}{a} \right\rfloor]$
- = $\sum_{y=0}^M n - \left\lfloor \frac{c*y + c - b - 1}{a} \right\rfloor$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩
展

- 所以原式 = $\sum_{y=0}^M \sum_{x=0}^n [x > \left\lfloor \frac{c*y + c - b - 1}{a} \right\rfloor]$
- = $\sum_{y=0}^M n - \left\lfloor \frac{c*y + c - b - 1}{a} \right\rfloor$
- = $n * (M + 1) - \sum_{y=0}^M \left\lfloor \frac{c*y + c - b - 1}{a} \right\rfloor$

洪华敦

需要用的一些数学符号

关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 所以原式 = $\sum_{y=0}^M \sum_{x=0}^n [x > \left\lfloor \frac{c*y + c - b - 1}{a} \right\rfloor]$
- = $\sum_{y=0}^M n - \left\lfloor \frac{c*y + c - b - 1}{a} \right\rfloor$
- = $n * (M + 1) - \sum_{y=0}^M \left\lfloor \frac{c*y + c - b - 1}{a} \right\rfloor$
- 于是我们的问题规模从 $(a\%c, c, b\%c, n)$ 变成了 $(c, a\%c, c - b\%c - 1, M)$

洪华敦

需要用到的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩
展

- 经过上面一番转化，我们的问题规模从 (a, c, b, n) 变成了 $(c, a\%c, c - b\%c - 1, M)$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算
法一些简单的扩
展

- 经过上面一番转化，我们的问题规模从 (a, c, b, n) 变成了 $(c, a\%c, c - b\%c - 1, M)$
- 显然当 a 为0时，我们可以很轻松地计算答案

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算
法一些简单的扩
展

- 经过上面一番转化，我们的问题规模从 (a, c, b, n) 变成了 $(c, a\%c, c - b\%c - 1, M)$
- 显然当 a 为0时，我们可以很轻松地计算答案
- 所以问题规模每次都从 (a, c) 变成 $(c, a\%c)$

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 经过上面一番转化，我们的问题规模从 (a, c, b, n) 变成了 $(c, a \% c, c - b \% c - 1, M)$
- 显然当 a 为0时，我们可以很轻松地计算答案
- 所以问题规模每次都从 (a, c) 变成 $(c, a \% c)$
- 这和计算 gcd 的时间复杂度是一样的

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 经过上面一番转化，我们的问题规模从 (a, c, b, n) 变成了 $(c, a \% c, c - b \% c - 1, M)$
- 显然当 a 为0时，我们可以很轻松地计算答案
- 所以问题规模每次都从 (a, c) 变成 $(c, a \% c)$
- 这和计算 gcd 的时间复杂度是一样的
- 所以时间复杂度是 $O(\log(a))$

洪华敦

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

洪华敦

需要用的的一
些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 简化题意：

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算
法一些简单的扩
展

- 简化题意：
- 给定 N, A, B, K, L , 计算：

$$\sum_{x=1}^n \binom{\lfloor \frac{Ax}{b} \rfloor}{K+1} \binom{n-x}{L}$$

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 简化题意:
- 给定 N, A, B, K, L , 计算:

$$\sum_{x=1}^n \binom{\lfloor \frac{Ax}{b} \rfloor}{K+1} \binom{n-x}{L}$$

- $N, A, B \leq 10^{18}, K, L \leq 10$

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 简化题意:
- 给定 N, A, B, K, L , 计算:

$$\sum_{x=1}^n \binom{\lfloor \frac{Ax}{b} \rfloor}{K+1} \binom{n-x}{L}$$

- $N, A, B \leq 10^{18}, K, L \leq 10$
- 答案对一个大质数取模

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 首先我们知道， $\binom{x}{k}$ 是一个关于x的k次多项式

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩
展

- 首先我们知道, $\binom{x}{k}$ 是一个关于 x 的 k 次多项式
- 设 $\binom{\lfloor \frac{Ax}{b} \rfloor}{K+1} = \sum_{i=0}^{K+1} c_i * (\lfloor \frac{Ax}{b} \rfloor)^i$

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 首先我们知道, $\binom{x}{k}$ 是一个关于 x 的 k 次多项式
- 设 $\binom{\lfloor \frac{Ax}{b} \rfloor}{K+1} = \sum_{i=0}^{K+1} c_i * (\lfloor \frac{Ax}{b} \rfloor)^i$
- 设 $\binom{n-x}{L} = \sum_{i=0}^L d_i * x^i$

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 首先我们知道, $\binom{x}{k}$ 是一个关于 x 的 k 次多项式
- 设 $\binom{\lfloor \frac{Ax}{b} \rfloor}{K+1} = \sum_{i=0}^{K+1} c_i * (\lfloor \frac{Ax}{b} \rfloor)^i$
- 设 $\binom{n-x}{L} = \sum_{i=0}^L d_i * x^i$
- 那么原式 = $\sum_{x=1}^n \sum_{i=0}^L d_i * x^i \sum_{j=0}^{K+1} c_j * (\lfloor \frac{Ax}{b} \rfloor)^j$

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 首先我们知道, $\binom{x}{k}$ 是一个关于 x 的 k 次多项式
- 设 $\binom{\lfloor \frac{Ax}{b} \rfloor}{K+1} = \sum_{i=0}^{K+1} c_i * (\lfloor \frac{Ax}{b} \rfloor)^i$
- 设 $\binom{n-x}{L} = \sum_{i=0}^L d_i * x^i$
- 那么原式 = $\sum_{x=1}^n \sum_{i=0}^L d_i * x^i \sum_{j=0}^{K+1} c_j * (\lfloor \frac{Ax}{b} \rfloor)^j$
- 等于 $\sum_{i=0}^L \sum_{j=0}^{K+1} d_i * c_j \sum_{x=1}^n x^i * (\lfloor \frac{Ax}{b} \rfloor)^j$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算
法一些简单的扩
展

- 首先我们知道, $\binom{x}{k}$ 是一个关于 x 的 k 次多项式
- 设 $\binom{\lfloor \frac{Ax}{b} \rfloor}{K+1} = \sum_{i=0}^{K+1} c_i * (\lfloor \frac{Ax}{b} \rfloor)^i$
- 设 $\binom{n-x}{L} = \sum_{i=0}^L d_i * x^i$
- 那么原式 = $\sum_{x=1}^n \sum_{i=0}^L d_i * x^i \sum_{j=0}^{K+1} c_j * (\lfloor \frac{Ax}{b} \rfloor)^j$
- 等于 $\sum_{i=0}^L \sum_{j=0}^{K+1} d_i * c_j \sum_{x=1}^n x^i * (\lfloor \frac{Ax}{b} \rfloor)^j$
- d_i 和 c_j 我们可以预处理用拉格朗日插值公式或者高斯消元算出来, 那么问题就变成了求后面那些东西

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用到的一些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 我们令 $f(a, c, b, i, j)$ 表示 $\sum_{x=0}^n x^i * (\lfloor \frac{a*x+b}{c} \rfloor)^j$

洪华敦

需要用的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们令 $f(a, c, b, i, j)$ 表示 $\sum_{x=0}^n x^i * (\lfloor \frac{a*x+b}{c} \rfloor)^j$
- 首先我们可以用二项式定理，让 $a, b < c$

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们令 $f(a, c, b, i, j)$ 表示 $\sum_{x=0}^n x^i * (\lfloor \frac{a*x+b}{c} \rfloor)^j$
- 首先我们可以用二项式定理，让 $a, b < c$
- 令 $M = \lfloor \frac{a*n+b}{c} \rfloor$

洪华敦

需要用的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们令 $f(a, c, b, i, j)$ 表示 $\sum_{x=0}^n x^i * (\left\lfloor \frac{ax+b}{c} \right\rfloor)^j$
- 首先我们可以用二项式定理，让 $a, b < c$
- 令 $M = \left\lfloor \frac{a*n+b}{c} \right\rfloor$
-

$$\sum_{x=0}^n x^U * (\left\lfloor \frac{a*x+b}{c} \right\rfloor)^V$$

洪华敦

需要用的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们令 $f(a, c, b, i, j)$ 表示 $\sum_{x=0}^n x^i * (\left\lfloor \frac{a*x+b}{c} \right\rfloor)^j$
- 首先我们可以用二项式定理，让 $a, b < c$
- 令 $M = \left\lfloor \frac{a*n+b}{c} \right\rfloor$
-

$$\sum_{x=0}^n x^U * \left(\left\lfloor \frac{a*x+b}{c} \right\rfloor \right)^V$$

-

$$= \sum_{x=0}^n x^U * \text{sum}_{y=0}^M [y < \left\lfloor \frac{a*x+b}{c} \right\rfloor] ((y+1)^V - y^V)$$

洪华敦

需要用的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们令 $f(a, c, b, i, j)$ 表示 $\sum_{x=0}^n x^i * (\left\lfloor \frac{a*x+b}{c} \right\rfloor)^j$
- 首先我们可以用二项式定理，让 $a, b < c$
- 令 $M = \left\lfloor \frac{a*n+b}{c} \right\rfloor$
-

$$\sum_{x=0}^n x^U * \left(\left\lfloor \frac{a*x+b}{c} \right\rfloor \right)^V$$

-

$$= \sum_{x=0}^n x^U * \text{sum}_{y=0}^M [y < \left\lfloor \frac{a*x+b}{c} \right\rfloor] ((y+1)^V - y^V)$$

-

$$= \sum_{y=0}^M ((y+1)^V - y^V) \sum_{x=0}^n x^U * [x > \left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor]$$

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 我们可以把 $(y + 1)^U - y^U$ 二项式展开

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们可以把 $(y + 1)^U - y^U$ 二项式展开
- 原式等于

洪华敦

需要用的的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们可以把 $(y + 1)^U - y^U$ 二项式展开
- 原式等于
- $= \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * \sum_{x=0}^n x^U * [x > \left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor]$

洪华敦

需要用的一些数学符号

关于欧几里得算
法类欧几里得算
法一些简单的扩
展

- 我们可以把 $(y + 1)^U - y^U$ 二项式展开
- 原式等于
- $= \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * \sum_{x=0}^n x^U * [x > \left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor]$
- $= \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * (S_U(n) - S_U(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor))$

洪华敦

需要用的一些数学符号

关于欧几里得算
法类欧几里得算
法一些简单的扩
展

- 我们可以把 $(y + 1)^U - y^U$ 二项式展开
- 原式等于
- $= \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * \sum_{x=0}^n x^U * [x > \left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor]$
- $= \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * (S_U(n) - S_U(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor))$
- $= \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * S_U(n) - \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * S_U(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor)$

需要用的一些数学符号

关于欧几里得算
法类欧几里得算
法一些简单的扩
展

- 我们可以把 $(y + 1)^U - y^U$ 二项式展开
- 原式等于
- $= \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * \sum_{x=0}^n x^U * [x > \left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor]$
- $= \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * (S_U(n) - S_U(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor))$
- $= \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * S_U(n) - \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * S_U(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor)$
- 前者等于 $\sum_{i=0}^{V-1} \binom{V}{i} S_i(M) * S_U(n)$, 于是问题就变成了求后者

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

洪华敦

需要用的一些数学符号

关于欧几里得算法

类欧几里得算法

一些简单的扩展

- 我们可以发现 $S_d(x)$ 是一个关于 x 的 $d + 1$ 次多项式

洪华敦

需要用的一些数学符号

关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们可以发现 $S_d(x)$ 是一个关于 x 的 $d + 1$ 次多项式
- 设 $S_d(x) = \sum_{i=0}^{d+1} a[d][i] * x^i$

需要用的一些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 我们可以发现 $S_d(x)$ 是一个关于 x 的 $d + 1$ 次多项式
- 设 $S_d(x) = \sum_{i=0}^{d+1} a[d][i] * x^i$
- 数组 a 可以用高斯消元预处理，或者直接带伯努利数进去

需要用的一些数学符号

关于欧几里得
算法

类欧几里得算
法

一些简单的扩
展

- 我们可以发现 $S_d(x)$ 是一个关于 x 的 $d + 1$ 次多项式
- 设 $S_d(x) = \sum_{i=0}^{d+1} a[d][i] * x^i$
- 数组 a 可以用高斯消元预处理，或者直接带伯努利数进去
- 原式等
$$\sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * \sum_{j=0}^{U+1} a[U][j] * \left(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor \right)^j$$

需要用的一些数学符号

关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们可以发现 $S_d(x)$ 是一个关于 x 的 $d + 1$ 次多项式
- 设 $S_d(x) = \sum_{i=0}^{d+1} a[d][i] * x^i$
- 数组 a 可以用高斯消元预处理，或者直接带伯努利数进去
- 原式等

$$\sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * \sum_{j=0}^{U+1} a[U][j] * \left(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor \right)^j$$
- 等于 $\sum_{i=0}^{V-1} \sum_{j=0}^{U+1} a[U][j] * \binom{V}{i} \sum_{y=0}^M y^i * \left(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor \right)^j$

需要用的一些数学符号

关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们可以发现 $S_d(x)$ 是一个关于 x 的 $d + 1$ 次多项式
- 设 $S_d(x) = \sum_{i=0}^{d+1} a[d][i] * x^i$
- 数组 a 可以用高斯消元预处理，或者直接带伯努利数进去
- 原式等

$$\sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * \sum_{j=0}^{U+1} a[U][j] * \left(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor\right)^j$$
- 等于 $\sum_{i=0}^{V-1} \sum_{j=0}^{U+1} a[U][j] * \binom{V}{i} \sum_{y=0}^M y^i * \left(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor\right)^j$
- 于是我们再次完成了 a, c 的互换，这样递归下去只有 $O(\log(A))$ 层

需要用的一些数学符号

关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们可以发现 $S_d(x)$ 是一个关于 x 的 $d + 1$ 次多项式
- 设 $S_d(x) = \sum_{i=0}^{d+1} a[d][i] * x^i$
- 数组 a 可以用高斯消元预处理，或者直接带伯努利数进去
- 原式等

$$\text{于 } \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * \sum_{j=0}^{U+1} a[U][j] * \left(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor \right)^j$$
- 等于 $\sum_{i=0}^{V-1} \sum_{j=0}^{U+1} a[U][j] * \binom{V}{i} \sum_{y=0}^M y^i * \left(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor \right)^j$
- 于是我们再次完成了 a, c 的互换，这样递归下去只有 $O(\log(A))$ 层
- 我们可以对于每个 (a, b, c, n) 求出对于所有 (i, j) 的答案，具体方法就是递归下去，然后用返回的那些答案更新

需要用的一
些数学符号关于欧几里得
算法类欧几里得算
法一些简单的扩
展

- 我们可以发现 $S_d(x)$ 是一个关于 x 的 $d + 1$ 次多项式
- 设 $S_d(x) = \sum_{i=0}^{d+1} a[d][i] * x^i$
- 数组 a 可以用高斯消元预处理，或者直接带伯努利数进去
- 原式等

$$\text{于 } \sum_{i=0}^{V-1} \binom{V}{i} \sum_{y=0}^M y^i * \sum_{j=0}^{U+1} a[U][j] * \left(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor \right)^j$$
- 等于 $\sum_{i=0}^{V-1} \sum_{j=0}^{U+1} a[U][j] * \binom{V}{i} \sum_{y=0}^M y^i * \left(\left\lfloor \frac{c*y+c-b-1}{a} \right\rfloor \right)^j$
- 于是我们再次完成了 a, c 的互换，这样递归下去只有 $O(\log(A))$ 层
- 我们可以对于每个 (a, b, c, n) 求出对于所有 (i, j) 的答案，具体方法就是递归下去，然后用返回的那些答案更新
- 用一些卷积优化可以把时间复杂度优化到 $O((K + L)^3 \log(A))$