Chia để trị I

Bùi Việt Dũng

Ngày 14 tháng 6 năm 2022

Hàm lô-ga-rit

Hàm lô-ga-rit

Định nghĩa

Cho hai số dương a, b, $a \neq 1$. Người ta chứng minh được tồn tại **duy nhất** một số $\alpha \in \mathbb{R}$ sao cho $a^\alpha = b$. Ta gọi số α đó là lô-ga-rít cơ số a của b và kí hiệu $\alpha = \log_a b$

Hàm lô-ga-rit

Định nghĩa

Cho hai số dương a, b, $a \neq 1$. Người ta chứng minh được tồn tại **duy nhất** một số $\alpha \in \mathbb{R}$ sao cho $a^\alpha = b$. Ta gọi số α đó là lô-ga-rít cơ số a của b và kí hiệu $\alpha = \log_a b$

Ví dụ:

Hàm lô-ga-rit

Định nghĩa

Cho hai số dương a, b, $a \neq 1$. Người ta chứng minh được tồn tại **duy nhất** một số $\alpha \in \mathbb{R}$ sao cho $a^\alpha = b$. Ta gọi số α đó là lô-ga-rít cơ số a của b và kí hiệu $\alpha = \log_a b$

Ví dụ:

- $2^5 = 32 \Leftrightarrow \log_2 32 = 5$

Hàm lô-ga-rit

Định nghĩa

Cho hai số dương a, b, $a \neq 1$. Người ta chứng minh được tồn tại **duy nhất** một số $\alpha \in \mathbb{R}$ sao cho $a^\alpha = b$. Ta gọi số α đó là lô-ga-rít cơ số a của b và kí hiệu $\alpha = \log_a b$

Ví dụ:

- $2^5 = 32 \Leftrightarrow \log_2 32 = 5$
- $3^3 = 27 \Leftrightarrow \log_3 27 = 2$

Hàm lô-ga-rit

Định nghĩa

Cho hai số dương a, b, $a \neq 1$. Người ta chứng minh được tồn tại **duy nhất** một số $\alpha \in \mathbb{R}$ sao cho $a^\alpha = b$. Ta gọi số α đó là lô-ga-rít cơ số a của b và kí hiệu $\alpha = \log_a b$

Ví dụ:

- $2^5 = 32 \Leftrightarrow \log_2 32 = 5$
- $3^3 = 27 \Leftrightarrow \log_3 27 = 2$
- $\log_5 625 = ?$

Hàm lô-ga-rit

Định nghĩa

Cho hai số dương a, b, $a \neq 1$. Người ta chứng minh được tồn tại **duy nhất** một số $\alpha \in \mathbb{R}$ sao cho $a^\alpha = b$. Ta gọi số α đó là lô-ga-rít cơ số a của b và kí hiệu $\alpha = \log_a b$

Ví dụ:

- $2^5 = 32 \Leftrightarrow \log_2 32 = 5$
- $3^3 = 27 \Leftrightarrow \log_3 27 = 2$
- $\log_5 625 = ?$
- $\log_{10} 10^5 = ?$

Hàm lô-ga-rit

Định nghĩa

Cho hai số dương a, b, $a \neq 1$. Người ta chứng minh được tồn tại **duy nhất** một số $\alpha \in \mathbb{R}$ sao cho $a^\alpha = b$. Ta gọi số α đó là lô-ga-rít cơ số a của b và kí hiệu $\alpha = \log_a b$

Ví dụ:

- $2^5 = 32 \Leftrightarrow \log_2 32 = 5$
- $3^3 = 27 \Leftrightarrow \log_3 27 = 2$
- $\log_5 625 = ?$
- $\log_{10} 10^5 = ?$
- $\log_a 1 = ?$

Hàm lô-ga-rit

Định nghĩa

Cho hai số dương a, b, $a \neq 1$. Người ta chứng minh được tồn tại **duy nhất** một số $\alpha \in \mathbb{R}$ sao cho $a^\alpha = b$. Ta gọi số α đó là lô-ga-rít cơ số a của b và kí hiệu $\alpha = \log_a b$

Ví dụ:

- $2^5 = 32 \Leftrightarrow \log_2 32 = 5$
- $3^3 = 27 \Leftrightarrow \log_3 27 = 2$
- $\log_5 625 = ?$
- $\log_{10} 10^5 = ?$
- $\log_a 1 = ?$

Với một số tài liệu $\log b = \log_{10} b$, với một số tài liệu khác, $\log b = \log_2 b$

Hàm lô-ga-rit

Định nghĩa

Cho hai số dương a, b, $a \neq 1$. Người ta chứng minh được tồn tại **duy nhất** một số $\alpha \in \mathbb{R}$ sao cho $a^\alpha = b$. Ta gọi số α đó là lô-ga-rít cơ số a của b và kí hiệu $\alpha = \log_a b$

Ví dụ:

- $2^5 = 32 \Leftrightarrow \log_2 32 = 5$
- $3^3 = 27 \Leftrightarrow \log_3 27 = 2$
- $\log_5 625 = ?$
- $\log_{10} 10^5 = ?$
- $\log_a 1 = ?$

Với một số tài liệu $\log b = \log_{10} b$, với một số tài liệu khác, $\log b = \log_2 b$

Khi $a = e \approx 2.72$, ta kí hiệu $\log_a b = \ln b$

Một số tính chất của lô-ga-rit

Một số tính chất của lô-ga-rit

Lô-ga-rit của một tích

$\log(ab) = \log a + \log b$

Một số tính chất của lô-ga-rit

Lô-ga-rit của một tích

$\log(ab) = \log a + \log b$

Hệ quả:

Một số tính chất của lô-ga-rit

Lô-ga-rit của một tích

$\log(ab) = \log a + \log b$

Hệ quả:

- $\log(a^n) = n \log a$

Một số tính chất của lô-ga-rit

Lô-ga-rit của một tích

$\log(ab) = \log a + \log b$

Hệ quả:

- $\log(a^n) = n \log a$
- $\log(\frac{a}{b}) = \log(ab^{-1}) = \log a - \log b$

Một số tính chất của lô-ga-rit

Lô-ga-rit của một tích

$\log(ab) = \log a + \log b$

Hệ quả:

- $\log(a^n) = n \log a$
- $\log(\frac{a}{b}) = \log(ab^{-1}) = \log a - \log b$

Đổi cơ số

Với $a \neq 1$ và $c \neq 1$, $\log_a b = \frac{\log_c b}{\log_c a}$

Một số tính chất của lô-ga-rit

Lô-ga-rit của một tích

$\log(ab) = \log a + \log b$

Hệ quả:

- $\log(a^n) = n \log a$
- $\log(\frac{a}{b}) = \log(ab^{-1}) = \log a - \log b$

Đổi cơ số

Với $a \neq 1$ và $c \neq 1$, $\log_a b = \frac{\log_c b}{\log_c a}$

So sánh

$\forall a > 1 : x < y \Leftrightarrow \log_a x < \log_a y$

Một số tính chất của lô-ga-rit

Lô-ga-rit của một tích

$\log(ab) = \log a + \log b$

Hệ quả:

- $\log(a^n) = n \log a$
- $\log(\frac{a}{b}) = \log(ab^{-1}) = \log a - \log b$

Đổi cơ số

Với $a \neq 1$ và $c \neq 1$, $\log_a b = \frac{\log_c b}{\log_c a}$

So sánh

$\forall a > 1 : x < y \Leftrightarrow \log_a x < \log_a y$

Thay vì so sánh x và y, ta có thể so sánh $\log x$ và $\log y$

Một số tính chất của lô-ga-rit

Một số tính chất của lô-ga-rit

Hàm log tăng rất chậm. Ví dụ với hàm $\log_2$:

Một số tính chất của lô-ga-rit

Hàm log tăng rất chậm. Ví dụ với hàm $\log_2$:

- $\log_2 10^3 < 10$ ($\log_2 1024 = 10$)

Một số tính chất của lô-ga-rit

Hàm log tăng rất chậm. Ví dụ với hàm $\log_2$:

- $\log_2 10^3 < 10$ ($\log_2 1024 = 10$)
- $\log_2 10^6 < 20$

Một số tính chất của lô-ga-rit

Hàm log tăng rất chậm. Ví dụ với hàm $\log_2$:

- $\log_2 10^3 < 10$ ($\log_2 1024 = 10$)
- $\log_2 10^6 < 20$
- $\log_2 10^9 < 30$

Một số tính chất của lô-ga-rit

Hàm log tăng rất chậm. Ví dụ với hàm $\log_2$:

- $\log_2 10^3 < 10$ ($\log_2 1024 = 10$)
- $\log_2 10^6 < 20$
- $\log_2 10^9 < 30$

Vì thế nên các thuật toán có độ phức tạp $O(\log n)$, $O(n \log n)$ thường có thời gian chạy nhanh.

$O(\log_2 n)$ hay $O(\log n)$?

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$
Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Lấy hàm $f(n) \in O(\log_a n)$ bất kì.

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$
Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Lấy hàm $f(n) \in O(\log_a n)$ bất kì. Ta cần chứng minh $f(n) \in O(\log n)$,

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$
Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Lấy hàm $f(n) \in O(\log_a n)$ bất kì. Ta cần chứng minh $f(n) \in O(\log n)$, tức $\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq c \log n$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$
Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Lấy hàm $f(n) \in O(\log_a n)$ bất kì. Ta cần chứng minh
$f(n) \in O(\log n)$, tức $\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq c \log n$
Do $f(n) \in O(\log_a n)$ nên

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$
Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Lấy hàm $f(n) \in O(\log_a n)$ bất kì. Ta cần chứng minh
$f(n) \in O(\log n)$, tức $\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq c \log n$
Do $f(n) \in O(\log_a n)$ nên
$\exists m_0, k > 0 : \forall n \geq m_0 : f(n) \leq k \log_a n$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Lấy hàm $f(n) \in O(\log_a n)$ bất kì. Ta cần chứng minh $f(n) \in O(\log n)$, tức $\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq c \log n$

Do $f(n) \in O(\log_a n)$ nên

$\exists m_0, k > 0 : \forall n \geq m_0 : f(n) \leq k \log_a n$

$\Rightarrow \exists m_0, k > 0 : \forall n \geq m_0 : f(n) \leq k \frac{\log n}{\log a}$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Lấy hàm $f(n) \in O(\log_a n)$ bất kì. Ta cần chứng minh $f(n) \in O(\log n)$, tức $\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq c \log n$

Do $f(n) \in O(\log_a n)$ nên

$\exists m_0, k > 0 : \forall n \geq m_0 : f(n) \leq k \log_a n$

$\Rightarrow \exists m_0, k > 0 : \forall n \geq m_0 : f(n) \leq k \frac{\log n}{\log a}$

$\Rightarrow \exists m_0, k > 0 : \forall n \geq m_0 : f(n) \leq \boxed{\frac{k}{\log a}} \log n$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho
$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$
Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho
$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$
Chọn $n_0 = m_0 > 0$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho
$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$
Chọn $n_0 = m_0 > 0$
Chọn $c = \frac{k}{\log a}$.

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho
$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$
Chọn $n_0 = m_0 > 0$
Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho

$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$

Chọn $n_0 = m_0 > 0$

Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$

Ta có $\forall n \geq n_0 : f(n) \leq c \log n$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho
$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$
Chọn $n_0 = m_0 > 0$
Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$
Ta có $\forall n \geq n_0 : f(n) \leq c \log n$
Do cách chọn n_0, c nên ta có thể kết luận

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho
$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$
Chọn $n_0 = m_0 > 0$
Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$
Ta có $\forall n \geq n_0 : f(n) \leq c \log n$
Do cách chọn n_0, c nên ta có thể kết luận
$\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq \log n \Rightarrow f(n) \in O(\log n)$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$
Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho
$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$
Chọn $n_0 = m_0 > 0$
Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$
Ta có $\forall n \geq n_0 : f(n) \leq c \log n$
Do cách chọn n_0, c nên ta có thể kết luận
$\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq \log n \Rightarrow f(n) \in O(\log n)$
Do ta lấy hàm $f(n) \in O(\log_a n)$ bất kì nên ta có

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$
Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho
$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$
Chọn $n_0 = m_0 > 0$
Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$
Ta có $\forall n \geq n_0 : f(n) \leq c \log n$
Do cách chọn n_0, c nên ta có thể kết luận
$\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq \log n \Rightarrow f(n) \in O(\log n)$
Do ta lấy hàm $f(n) \in O(\log_a n)$ bất kì nên ta có
$\forall f(n) \in O(\log_a n) : f(n) \in O(\log n)$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho

$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$

Chọn $n_0 = m_0 > 0$

Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$

Ta có $\forall n \geq n_0 : f(n) \leq c \log n$

Do cách chọn n_0, c nên ta có thể kết luận

$\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq \log n \Rightarrow f(n) \in O(\log n)$

Do ta lấy hàm $f(n) \in O(\log_a n)$ bất kì nên ta có

$\forall f(n) \in O(\log_a n) : f(n) \in O(\log n) \Rightarrow O(\log_a n) \subset O(\log n)$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho

$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$

Chọn $n_0 = m_0 > 0$

Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$

Ta có $\forall n \geq n_0 : f(n) \leq c \log n$

Do cách chọn n_0, c nên ta có thể kết luận

$\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq \log n \Rightarrow f(n) \in O(\log n)$

Do ta lấy hàm $f(n) \in O(\log_a n)$ bất kì nên ta có

$\forall f(n) \in O(\log_a n) : f(n) \in O(\log n) \Rightarrow O(\log_a n) \subset O(\log n)$

Tương tự, ta cũng chứng minh được $O(\log n) \subset O(\log_a n)$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho

$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$

Chọn $n_0 = m_0 > 0$

Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$

Ta có $\forall n \geq n_0 : f(n) \leq c \log n$

Do cách chọn n_0, c nên ta có thể kết luận

$\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq \log n \Rightarrow f(n) \in O(\log n)$

Do ta lấy hàm $f(n) \in O(\log_a n)$ bất kì nên ta có

$\forall f(n) \in O(\log_a n) : f(n) \in O(\log n) \Rightarrow O(\log_a n) \subset O(\log n)$

Tương tự, ta cũng chứng minh được $O(\log n) \subset O(\log_a n)$ và vì thế mà $O(\log_a n) = O(\log n)$

$O(\log_2 n)$ hay $O(\log n)$?

Cơ số không quan trọng

Chứng minh rằng: $O(\log_a n) = O(\log n)$ với mọi $a > 1$

Trong phần lớn trường hợp, cơ số của hàm lô-ga-rit không quan trọng khi phân tích độ phức tạp tính toán

Như vậy ta có thể chọn các số $m_0, k > 0$ sao cho

$\forall n \geq m_0 : f(n) \leq \frac{k}{\log a} \log n$

Chọn $n_0 = m_0 > 0$

Chọn $c = \frac{k}{\log a}$. Do $k > 0$ và $(a > 1 \Rightarrow \log a > 0)$ nên $c > 0$

Ta có $\forall n \geq n_0 : f(n) \leq c \log n$

Do cách chọn n_0, c nên ta có thể kết luận

$\exists n_0, c > 0 : \forall n \geq n_0 : f(n) \leq \log n \Rightarrow f(n) \in O(\log n)$

Do ta lấy hàm $f(n) \in O(\log_a n)$ bất kì nên ta có

$\forall f(n) \in O(\log_a n) : f(n) \in O(\log n) \Rightarrow O(\log_a n) \subset O(\log n)$

Tương tự, ta cũng chứng minh được $O(\log n) \subset O(\log_a n)$ và vì thế mà $O(\log_a n) = O(\log n)$

Đếm số chữ số

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

- Tính chính xác $n!$ bằng thuật toán tính toán số lớn?

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

- Tính chính xác $n!$ bằng thuật toán tính toán số lớn?
  - $10^5!$ có ít nhất 10^5 chữ số.

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

- Tính chính xác $n!$ bằng thuật toán tính toán số lớn?
  - $10^5!$ có ít nhất 10^5 chữ số.
  - Từ $10^5!$ đến $10^6!$ ta cần thực hiện $(9 \times 10^5)!$ phép tính nhân, mỗi phép tính tốn ít nhất 10^5 phép tính $\Rightarrow$ quá thời gian.

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

- Tính chính xác $n!$ bằng thuật toán tính toán số lớn?
  - $10^5!$ có ít nhất 10^5 chữ số.
  - Từ $10^5!$ đến $10^6!$ ta cần thực hiện $(9 \times 10^5)!$ phép tính nhân, mỗi phép tính tốn ít nhất 10^5 phép tính $\Rightarrow$ quá thời gian.
- Dùng `long double` để tính gần đúng $(n!)$?

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

- Tính chính xác $n!$ bằng thuật toán tính toán số lớn?
  - $10^5!$ có ít nhất 10^5 chữ số.
  - Từ $10^5!$ đến $10^6!$ ta cần thực hiện $(9 \times 10^5)!$ phép tính nhân, mỗi phép tính tốn ít nhất 10^5 phép tính $\Rightarrow$ quá thời gian.
- Dùng `long double` để tính gần đúng $(n!)$?
  - Kiểu `long double` có giới hạn là $\approx 10^{4932}$

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

Giả sử $n!$ có k chữ số

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

Giả sử $n!$ có k chữ số thì $10^{k-1} \leq n! < 10^k$

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

Giả sử $n!$ có k chữ số thì $10^{k-1} \leq n! < 10^k$
$\Rightarrow \log 10^{k-1} \leq \log n! < 10^k$

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

Giả sử $n!$ có k chữ số thì $10^{k-1} \leq n! < 10^k$

$\Rightarrow \log 10^{k-1} \leq \log n! < 10^k$

$\Rightarrow k - 1 \leq \log 1 + \log 2 + \log 3 + ... + \log n < k$

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

Giả sử $n!$ có k chữ số thì $10^{k-1} \leq n! < 10^k$

$\Rightarrow \log 10^{k-1} \leq \log n! < 10^k$

$\Rightarrow k - 1 \leq \log 1 + \log 2 + \log 3 + ... + \log n < k$

Như vậy là ta chỉ cần tính $\log 1 + \log 2 + \log 3 + ... \log n$ trong $O(n)$

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

Giả sử $n!$ có k chữ số thì $10^{k-1} \leq n! < 10^k$

$\Rightarrow \log 10^{k-1} \leq \log n! < 10^k$

$\Rightarrow k - 1 \leq \log 1 + \log 2 + \log 3 + ... + \log n < k$

Như vậy là ta chỉ cần tính $\log 1 + \log 2 + \log 3 + ... \log n$ trong $O(n)$, rồi làm tròn kết quả xuống là tính ra $k - 1$

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

Giả sử $n!$ có k chữ số thì $10^{k-1} \leq n! < 10^k$

$\Rightarrow \log 10^{k-1} \leq \log n! < 10^k$

$\Rightarrow k - 1 \leq \log 1 + \log 2 + \log 3 + ... + \log n < k$

Như vậy là ta chỉ cần tính $\log 1 + \log 2 + \log 3 + ... \log n$ trong $O(n)$, rồi làm tròn kết quả xuống là tính ra $k - 1 \rightarrow$ tính ra k.

Đếm số chữ số

Bài toán

Đếm số chữ số của $n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n$ $(n \leq 10^6)$

Ý tưởng

Giả sử $n!$ có k chữ số thì $10^{k-1} \leq n! < 10^k$
$\Rightarrow \log 10^{k-1} \leq \log n! < 10^k$
$\Rightarrow k - 1 \leq \log 1 + \log 2 + \log 3 + ... + \log n < k$
Như vậy là ta chỉ cần tính $\log 1 + \log 2 + \log 3 + ... \log n$ trong $O(n)$, rồi làm tròn kết quả xuống là tính ra $k - 1 \rightarrow$ tính ra k.

Mở rộng bài toán

- Đếm số chữ số của $n!$ $(n \leq 10^6)$ khi viết $n!$ ở hệ cơ số k

Chia để trị

Để giải một bài toán, ta có thể thực hiện các bước sau:

Chia để trị

Để giải một bài toán, ta có thể thực hiện các bước sau:

1. Nếu bài toán đủ đơn giản, ta giải luôn.

Chia để trị

Để giải một bài toán, ta có thể thực hiện các bước sau:

1. Nếu bài toán đủ đơn giản, ta giải luôn.
2. Chia bài toán thành các bài toán con.

Chia để trị

Để giải một bài toán, ta có thể thực hiện các bước sau:

1. Nếu bài toán đủ đơn giản, ta giải luôn.
2. Chia bài toán thành các bài toán con.
3. (*Tùy vào bài toán*) Xác định bài toán con nào cần giải, bài toán con nào không.

Chia để trị

Để giải một bài toán, ta có thể thực hiện các bước sau:

1. Nếu bài toán đủ đơn giản, ta giải luôn.
2. Chia bài toán thành các bài toán con.
3. (*Tùy vào bài toán*) Xác định bài toán con nào cần giải, bài toán con nào không.
4. Giải các bài toán con cần giải (bằng cách chia tiếp các bài toán con này hoặc giải trực tiếp nếu bài toán con đủ nhỏ)

Chia để trị

Để giải một bài toán, ta có thể thực hiện các bước sau:

1. Nếu bài toán đủ đơn giản, ta giải luôn.
2. Chia bài toán thành các bài toán con.
3. (*Tùy vào bài toán*) Xác định bài toán con nào cần giải, bài toán con nào không.
4. Giải các bài toán con cần giải (bằng cách chia tiếp các bài toán con này hoặc giải trực tiếp nếu bài toán con đủ nhỏ)
5. Kết hợp đáp án của các bài toán con để ra được đáp án của bài toán lớn.

Chia để trị

Để giải một bài toán, ta có thể thực hiện các bước sau:

1. Nếu bài toán đủ đơn giản, ta giải luôn.
2. Chia bài toán thành các bài toán con.
3. (*Tùy vào bài toán*) Xác định bài toán con nào cần giải, bài toán con nào không.
4. Giải các bài toán con cần giải (bằng cách chia tiếp các bài toán con này hoặc giải trực tiếp nếu bài toán con đủ nhỏ)
5. Kết hợp đáp án của các bài toán con để ra được đáp án của bài toán lớn.

Phương pháp giải bài trên được gọi là phương pháp **Chia để trị**

Tìm kiếm nhị phân

Cho một bài toán có các nghiệm là các số nằm trong $[l, r]$ thỏa mãn điều kiện sau:

Tìm kiếm nhị phân

Cho một bài toán có các nghiệm là các số nằm trong $[l, r]$ thỏa mãn điều kiện sau:

- Nếu x là nghiệm của bài toán thì $\forall y \leq x : y$ cũng là nghiệm của bài toán.

Tìm kiếm nhị phân

Cho một bài toán có các nghiệm là các số nằm trong $[l, r]$ thỏa mãn điều kiện sau:

- Nếu x là nghiệm của bài toán thì $\forall y \leq x : y$ cũng là nghiệm của bài toán.
- Nếu x không là nghiệm của bài toán thì $\forall y \geq x : y$ cũng không là nghiệm của bài toán.

Tìm kiếm nhị phân

Cho một bài toán có các nghiệm là các số nằm trong $[l, r]$ thỏa mãn điều kiện sau:

- Nếu x là nghiệm của bài toán thì $\forall y \leq x : y$ cũng là nghiệm của bài toán.
- Nếu x không là nghiệm của bài toán thì $\forall y \geq x : y$ cũng không là nghiệm của bài toán.
- Bài toán tồn tại nghiệm lớn nhất.

Tìm kiếm nhị phân

Cho một bài toán có các nghiệm là các số nằm trong $[l, r]$ thỏa mãn điều kiện sau:

- Nếu x là nghiệm của bài toán thì $\forall y \leq x : y$ cũng là nghiệm của bài toán.
- Nếu x không là nghiệm của bài toán thì $\forall y \geq x : y$ cũng không là nghiệm của bài toán.
- Bài toán tồn tại nghiệm lớn nhất.

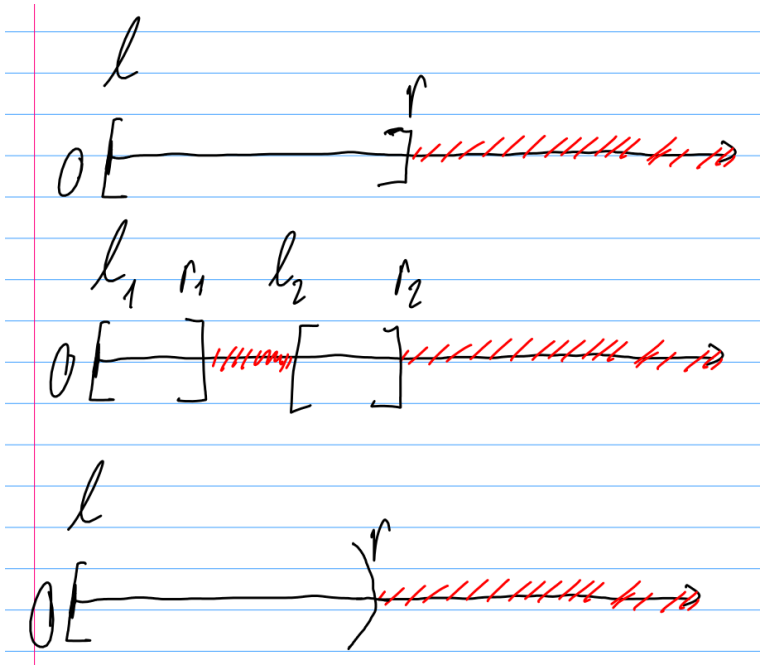
Hãy tìm nghiệm lớn nhất của bài toán.

Luyện tập

Giả sử tập nghiệm của một bài toán nào đó được biểu diễn như trong hình dưới:

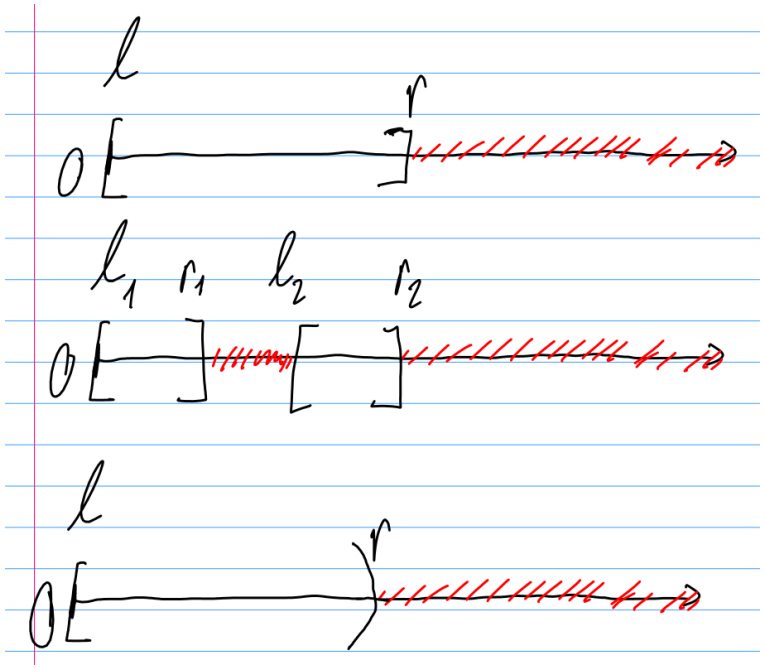
Luyện tập

Giả sử tập nghiệm của một bài toán nào đó được biểu diễn như trong hình dưới:



Luyện tập

Giả sử tập nghiệm của một bài toán nào đó được biểu diễn như trong hình dưới:



Ta có thể sử dụng thuật toán tìm kiếm nhị phân để tìm nghiệm lớn nhất của những bài toán nào?

Bài toán

Bài toán

Tính chính xác giá trị của $\lfloor \sqrt{n} \rfloor$ với n là một số nguyên dương lớn.

Bài toán

Bài toán

Tính chính xác giá trị của $\lfloor \sqrt{n} \rfloor$ với n là một số nguyên dương lớn.

Ý tưởng

- Dùng hàm `floor(sqrt(n))` trong C++?

Bài toán

Bài toán

Tính chính xác giá trị của $\lfloor \sqrt{n} \rfloor$ với n là một số nguyên dương lớn.

Ý tưởng

- Dùng hàm `floor(sqrt(n))` trong C++?
  - Chỉ đúng với $n \leq 10^{14}$.

Bài toán

Bài toán

Tính chính xác giá trị của $\lfloor \sqrt{n} \rfloor$ với n là một số nguyên dương lớn.

Ý tưởng

- Dùng hàm `floor(sqrt(n))` trong C++?
  - Chỉ đúng với $n \leq 10^{14}$.
  - Cách sửa (với $n \leq 8 \times 10^{18}$):

Bài toán

Bài toán

Tính chính xác giá trị của $\lfloor \sqrt{n} \rfloor$ với n là một số nguyên dương lớn.

Ý tưởng

- Dùng hàm `floor(sqrt(n))` trong C++?
  - Chỉ đúng với $n \leq 10^{14}$.
  - Cách sửa (với $n \leq 8 \times 10^{18}$):
    - Kiểm tra hết các số nguyên trong $[\sqrt{n} - 3, \sqrt{n} + 3]$. Hạn chế: tăng hằng số, cần mở rộng khoảng này nếu n lớn hơn.

Bài toán

Bài toán

Tính chính xác giá trị của $\lfloor \sqrt{n} \rfloor$ với n là một số nguyên dương lớn.

Ý tưởng

- Dùng hàm `floor(sqrt(n))` trong C++?
  - Chỉ đúng với $n \leq 10^{14}$.
  - Cách sửa (với $n \leq 8 \times 10^{18}$):
    - Kiểm tra hết các số nguyên trong $[\sqrt{n} - 3, \sqrt{n} + 3]$. Hạn chế: tăng hằng số, cần mở rộng khoảng này nếu n lớn hơn.
    - Chuyển n sang `long double` trước khi dùng hàm `sqrt`

Bài toán

Bài toán

Tính chính xác giá trị của $\lfloor \sqrt{n} \rfloor$ với n là một số nguyên dương lớn.

Ý tưởng

- Dùng hàm `floor(sqrt(n))` trong C++?
  - Chỉ đúng với $n \leq 10^{14}$.
  - Cách sửa (với $n \leq 8 \times 10^{18}$):
    - Kiểm tra hết các số nguyên trong $[\sqrt{n} - 3, \sqrt{n} + 3]$. Hạn chế: tăng hằng số, cần mở rộng khoảng này nếu n lớn hơn.
    - Chuyển n sang `long double` trước khi dùng hàm `sqrt`
    - Dùng hàm `sqrtl`

Bài toán

Bài toán

Tính chính xác giá trị của $\lfloor \sqrt{n} \rfloor$ với n là một số nguyên dương lớn.

Ý tưởng

- Dùng hàm `floor(sqrt(n))` trong C++?
  - Chỉ đúng với $n \leq 10^{14}$.
  - Cách sửa (với $n \leq 8 \times 10^{18}$):
    - Kiểm tra hết các số nguyên trong $[\sqrt{n} - 3, \sqrt{n} + 3]$. Hạn chế: tăng hằng số, cần mở rộng khoảng này nếu n lớn hơn.
    - Chuyển n sang `long double` trước khi dùng hàm `sqrt`
    - Dùng hàm `sqrtl`
- Dùng hàm `math.isqrt(n)` trong Python?

Kiểm tra điều kiện

Kiểm tra điều kiện

Ta gọi bài toán hiện tại là "Tìm các số nguyên dương nhỏ hơn hoặc bằng $\sqrt{n}$". Do $1 \leq \sqrt{n} \leq n$, ta biết nghiệm của bài toán nằm trong $[1, n]$ nên ta có thể thêm điều kiện này vào bài toán mà không làm thay đổi tập nghiệm.

- Giả sử x là nghiệm của bài toán,

Kiểm tra điều kiện

Ta gọi bài toán hiện tại là "Tìm các số nguyên dương nhỏ hơn hoặc bằng $\sqrt{n}$". Do $1 \leq \sqrt{n} \leq n$, ta biết nghiệm của bài toán nằm trong $[1, n]$ nên ta có thể thêm điều kiện này vào bài toán mà không làm thay đổi tập nghiệm.

- Giả sử x là nghiệm của bài toán, tức $x \leq \sqrt{n}$.

Kiểm tra điều kiện

Ta gọi bài toán hiện tại là "Tìm các số nguyên dương nhỏ hơn hoặc bằng $\sqrt{n}$". Do $1 \leq \sqrt{n} \leq n$, ta biết nghiệm của bài toán nằm trong $[1, n]$ nên ta có thể thêm điều kiện này vào bài toán mà không làm thay đổi tập nghiệm.

- Giả sử x là nghiệm của bài toán, tức $x \leq \sqrt{n}$. Chọn $y \leq x$ bất kì.

Kiểm tra điều kiện

Ta gọi bài toán hiện tại là "Tìm các số nguyên dương nhỏ hơn hoặc bằng $\sqrt{n}$". Do $1 \leq \sqrt{n} \leq n$, ta biết nghiệm của bài toán nằm trong $[1, n]$ nên ta có thể thêm điều kiện này vào bài toán mà không làm thay đổi tập nghiệm.

- Giả sử x là nghiệm của bài toán, tức $x \leq \sqrt{n}$. Chọn $y \leq x$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ y \leq x \end{cases} \Rightarrow y \leq \sqrt{n}$

Kiểm tra điều kiện

Ta gọi bài toán hiện tại là "Tìm các số nguyên dương nhỏ hơn hoặc bằng $\sqrt{n}$". Do $1 \leq \sqrt{n} \leq n$, ta biết nghiệm của bài toán nằm trong $[1, n]$ nên ta có thể thêm điều kiện này vào bài toán mà không làm thay đổi tập nghiệm.

- Giả sử x là nghiệm của bài toán, tức $x \leq \sqrt{n}$. Chọn $y \leq x$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ y \leq x \end{cases} \Rightarrow y \leq \sqrt{n}$ nên y cũng là nghiệm của bài toán.

Kiểm tra điều kiện

Ta gọi bài toán hiện tại là "Tìm các số nguyên dương nhỏ hơn hoặc bằng $\sqrt{n}$". Do $1 \leq \sqrt{n} \leq n$, ta biết nghiệm của bài toán nằm trong $[1, n]$ nên ta có thể thêm điều kiện này vào bài toán mà không làm thay đổi tập nghiệm.

- Giả sử x là nghiệm của bài toán, tức $x \leq \sqrt{n}$. Chọn $y \leq x$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ y \leq x \end{cases} \Rightarrow y \leq \sqrt{n}$ nên y cũng là nghiệm của bài toán.
- Ta cũng chứng minh được nếu x không là nghiệm của bài toán thì $\forall y \geq x : y$ cũng không là nghiệm của bài toán.

Kiểm tra điều kiện

Ta gọi bài toán hiện tại là "Tìm các số nguyên dương nhỏ hơn hoặc bằng $\sqrt{n}$". Do $1 \leq \sqrt{n} \leq n$, ta biết nghiệm của bài toán nằm trong $[1, n]$ nên ta có thể thêm điều kiện này vào bài toán mà không làm thay đổi tập nghiệm.

- Giả sử x là nghiệm của bài toán, tức $x \leq \sqrt{n}$. Chọn $y \leq x$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ y \leq x \end{cases} \Rightarrow y \leq \sqrt{n}$ nên y cũng là nghiệm của bài toán.
- Ta cũng chứng minh được nếu x không là nghiệm của bài toán thì $\forall y \geq x : y$ cũng không là nghiệm của bài toán.

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương,

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$

  Chọn $x \in S$ bất kì.

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$
  Chọn $x \in S$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ \sqrt{n} \leq n \end{cases} \Rightarrow x \leq n.$

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$
  Chọn $x \in S$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ \sqrt{n} \leq n \end{cases} \Rightarrow x \leq n$. Do đó
  $\forall x \in S : x \leq n$

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$
  Chọn $x \in S$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ \sqrt{n} \leq n \end{cases} \Rightarrow x \leq n$. Do đó
  $\forall x \in S : x \leq n$
  Xét tập $S' = \{n - x | x \in S\}$.

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$
  Chọn $x \in S$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ \sqrt{n} \leq n \end{cases} \Rightarrow x \leq n$. Do đó
  $\forall x \in S : x \leq n$
  Xét tập $S' = \{n - x | x \in S\}$. Đây là tập con của tập $\mathbb{N}$, và $S' \neq \emptyset$ (do $S \neq \emptyset$)

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$
  Chọn $x \in S$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ \sqrt{n} \leq n \end{cases} \Rightarrow x \leq n$. Do đó
  $\forall x \in S : x \leq n$
  Xét tập $S' = \{n - x | x \in S\}$. Đây là tập con của tập $\mathbb{N}$, và $S' \neq \emptyset$ (do $S \neq \emptyset$)nên $\min S'$ tồn tại và ta có thể chọn $y \in S'$ sao cho $y = \min S'$

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$
  Chọn $x \in S$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ \sqrt{n} \leq n \end{cases} \Rightarrow x \leq n$. Do đó
  $\forall x \in S : x \leq n$
  Xét tập $S' = \{n - x | x \in S\}$. Đây là tập con của tập $\mathbb{N}$, và $S' \neq \emptyset$ (do $S \neq \emptyset$)nên $\min S'$ tồn tại và ta có thể chọn $y \in S'$ sao cho $y = \min S'$
  Do $y \in S$ nên ta có thể chọn $x \in S$ sao cho
  $y = n - x \Rightarrow x = n - y = n - \min S'$.

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$
  Chọn $x \in S$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ \sqrt{n} \leq n \end{cases} \Rightarrow x \leq n$. Do đó
  $\forall x \in S : x \leq n$
  Xét tập $S' = \{n - x | x \in S\}$. Đây là tập con của tập $\mathbb{N}$, và $S' \neq \emptyset$ (do $S \neq \emptyset$)nên $\min S'$ tồn tại và ta có thể chọn $y \in S'$ sao cho $y = \min S'$
  Do $y \in S$ nên ta có thể chọn $x \in S$ sao cho
  $y = n - x \Rightarrow x = n - y = n - \min S'$.
  Ta có thể chứng minh được $x = \max S$.

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$
  Chọn $x \in S$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ \sqrt{n} \leq n \end{cases} \Rightarrow x \leq n$. Do đó
  $\forall x \in S : x \leq n$
  Xét tập $S' = \{n - x | x \in S\}$. Đây là tập con của tập $\mathbb{N}$, và $S' \neq \emptyset$ (do $S \neq \emptyset$)nên $\min S'$ tồn tại và ta có thể chọn $y \in S'$ sao cho $y = \min S'$
  Do $y \in S$ nên ta có thể chọn $x \in S$ sao cho
  $y = n - x \Rightarrow x = n - y = n - \min S'$.
  Ta có thể chứng minh được $x = \max S$. Vì vậy bài toán có nghiệm lớn nhất.

Kiểm tra điều kiện

- Gọi S là tập nghiệm của bài toán.
  Theo yêu cầu bài toán, $S \subset \mathbb{N}$
  Với n nguyên dương, $1 \leq n \Rightarrow 1 \leq \sqrt{n}$ nên $1 \in S \Rightarrow S \neq \emptyset$
  Chọn $x \in S$ bất kì. Ta có $\begin{cases} x \leq \sqrt{n} \\ \sqrt{n} \leq n \end{cases} \Rightarrow x \leq n$. Do đó
  $\forall x \in S : x \leq n$
  Xét tập $S' = \{n - x | x \in S\}$. Đây là tập con của tập $\mathbb{N}$, và $S' \neq \emptyset$ (do $S \neq \emptyset$)nên $\min S'$ tồn tại và ta có thể chọn $y \in S'$ sao cho $y = \min S'$
  Do $y \in S$ nên ta có thể chọn $x \in S$ sao cho
  $y = n - x \Rightarrow x = n - y = n - \min S'$.
  Ta có thể chứng minh được $x = \max S$. Vì vậy bài toán có nghiệm lớn nhất.

Từ ba điều đã chứng minh ở trên, ta biết được rằng ta có thể sử dụng thuật toán tìm kiếm nhị phân để tìm nghiệm lớn nhất của bài toán.

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con:

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m + 1, r]$ (bằng $f(m + 1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m + 1, r]$ (bằng $f(m + 1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$
- Nếu m không là nghiệm của bài toán thì:

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m + 1, r]$ (bằng $f(m + 1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$
- Nếu m không là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[m + 1, r]$

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m + 1, r]$ (bằng $f(m + 1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$
- Nếu m không là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[m + 1, r]$
  - Đáp án của bài toán trên $[l, r]$ là $f(l, m)$

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m + 1, r]$ (bằng $f(m + 1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$
- Nếu m không là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[m + 1, r]$
  - Đáp án của bài toán trên $[l, r]$ là $f(l, m)$
- Nếu m là nghiệm của bài toán thì:

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m + 1, r]$ (bằng $f(m + 1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$
- Nếu m không là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[m + 1, r]$
  - Đáp án của bài toán trên $[l, r]$ là $f(l, m)$
- Nếu m là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[l, m]$

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m + 1, r]$ (bằng $f(m + 1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$
- Nếu m không là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[m + 1, r]$
  - Đáp án của bài toán trên $[l, r]$ là $f(l, m)$
- Nếu m là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[l, m]$
  - Đáp án của bài toán trên $[l, r]$ là

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m+1, r]$ (bằng $f(m+1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$
- Nếu m không là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[m+1, r]$
  - Đáp án của bài toán trên $[l, r]$ là $f(l, m)$
- Nếu m là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[l, m]$
  - Đáp án của bài toán trên $[l, r]$ là
    - $f(m+1, r)$ nếu bài toán có nghiệm trên $[m+1, r]$

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m+1, r]$ (bằng $f(m+1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$
- Nếu m không là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[m+1, r]$
  - Đáp án của bài toán trên $[l, r]$ là $f(l, m)$
- Nếu m là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[l, m]$
  - Đáp án của bài toán trên $[l, r]$ là
    - $f(m+1, r)$ nếu bài toán có nghiệm trên $[m+1, r]$
    - m nếu bài toán trên $[m+1, r]$ vô nghiệm.

Cách làm

Giả sử ta có hàm $f(l, r)$ cho biết nghiệm lớn nhất nếu bài toán được giải trên $[l, r]$.

- Trường hợp đơn giản: $l = r$. Khi đó, ta chỉ cần kiểm tra $l \leq \sqrt{n}$ hay không là giải được bài toán.
- Ta có thể chia bài toán thành hai bài toán con: Nghiệm lớn nhất trong $[l, m]$ (bằng $f(l, m)$) và nghiệm lớn nhất trong $[m+1, r]$ (bằng $f(m+1, r)$) với $m = \lfloor \frac{l+r}{2} \rfloor$
- Nếu m không là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[m+1, r]$
  - Đáp án của bài toán trên $[l, r]$ là $f(l, m)$
- Nếu m là nghiệm của bài toán thì:
  - Ta không cần thiết phải giải bài toán trên $[l, m]$
  - Đáp án của bài toán trên $[l, r]$ là
    - $f(m+1, r)$ nếu bài toán có nghiệm trên $[m+1, r]$
    - m nếu bài toán trên $[m+1, r]$ vô nghiệm.

Đáp án của bài toán ta cần giải là $f(1, n)$

Tính dừng của thuật toán

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

- $A(0)$ đúng.

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

- $A(0)$ đúng.
- Giả sử $A(0), A(1), ..., A(k)$ đều đúng.

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

- $A(0)$ đúng.
- Giả sử $A(0), A(1), ..., A(k)$ đều đúng. Ta cần chứng minh $A(k+1)$ đúng.

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

- $A(0)$ đúng.
- Giả sử $A(0), A(1), ..., A(k)$ đều đúng. Ta cần chứng minh $A(k+1)$ đúng.
  Thuật toán lúc đầu sẽ chia bài toán trên đoạn độ dài $k+1$ thành hai bài toán con với độ dài $\lfloor \frac{k+1}{2} \rfloor$ và $k - \lfloor \frac{k+1}{2} \rfloor$

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

- $A(0)$ đúng.
- Giả sử $A(0), A(1), ..., A(k)$ đều đúng. Ta cần chứng minh $A(k+1)$ đúng.
  Thuật toán lúc đầu sẽ chia bài toán trên đoạn độ dài $k+1$ thành hai bài toán con với độ dài $\lfloor \frac{k+1}{2} \rfloor$ và $k - \lfloor \frac{k+1}{2} \rfloor$. Dễ thấy hai số này là các số nguyên không âm nhỏ hơn $k+1$, nên $A(\lfloor \frac{k+1}{2} \rfloor)$ và $A(k - \lfloor \frac{k+1}{2} \rfloor)$ đúng

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

- $A(0)$ đúng.
- Giả sử $A(0), A(1), ..., A(k)$ đều đúng. Ta cần chứng minh $A(k+1)$ đúng.
  Thuật toán lúc đầu sẽ chia bài toán trên đoạn độ dài $k+1$ thành hai bài toán con với độ dài $\lfloor \frac{k+1}{2} \rfloor$ và $k - \lfloor \frac{k+1}{2} \rfloor$. Dễ thấy hai số này là các số nguyên không âm nhỏ hơn $k+1$, nên $A(\lfloor \frac{k+1}{2} \rfloor)$ và $A(k - \lfloor \frac{k+1}{2} \rfloor)$ đúng . Do đó ta sẽ có thể chia tiếp bài toán con trên đoạn độ dài $\lfloor \frac{k+1}{2} \rfloor$ và trên đoạn độ dài $k - \lfloor \frac{k+1}{2} \rfloor$ thành các bài toán con, mỗi bài toán trên đoạn độ dài 0

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

- $A(0)$ đúng.
- Giả sử $A(0), A(1), ..., A(k)$ đều đúng. Ta cần chứng minh $A(k+1)$ đúng.
  Thuật toán lúc đầu sẽ chia bài toán trên đoạn độ dài $k+1$ thành hai bài toán con với độ dài $\lfloor \frac{k+1}{2} \rfloor$ và $k - \lfloor \frac{k+1}{2} \rfloor$. Dễ thấy hai số này là các số nguyên không âm nhỏ hơn $k+1$, nên $A(\lfloor \frac{k+1}{2} \rfloor)$ và $A(k - \lfloor \frac{k+1}{2} \rfloor)$ đúng . Do đó ta sẽ có thể chia tiếp bài toán con trên đoạn độ dài $\lfloor \frac{k+1}{2} \rfloor$ và trên đoạn độ dài $k - \lfloor \frac{k+1}{2} \rfloor$ thành các bài toán con, mỗi bài toán trên đoạn độ dài 0 . Kết hợp hai kết quả chia bài toán thì ta thấy bài toán trên đoạn độ dài $k+1$ được chia thành các bài toán con, mỗi bài toán trên đoạn độ dài 0

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

- $A(0)$ đúng.
- Giả sử $A(0), A(1), ..., A(k)$ đều đúng. Ta cần chứng minh $A(k+1)$ đúng.
  Thuật toán lúc đầu sẽ chia bài toán trên đoạn độ dài $k+1$ thành hai bài toán con với độ dài $\lfloor \frac{k+1}{2} \rfloor$ và $k - \lfloor \frac{k+1}{2} \rfloor$. Dễ thấy hai số này là các số nguyên không âm nhỏ hơn $k+1$, nên $A(\lfloor \frac{k+1}{2} \rfloor)$ và $A(k - \lfloor \frac{k+1}{2} \rfloor)$ đúng . Do đó ta sẽ có thể chia tiếp bài toán con trên đoạn độ dài $\lfloor \frac{k+1}{2} \rfloor$ và trên đoạn độ dài $k - \lfloor \frac{k+1}{2} \rfloor$ thành các bài toán con, mỗi bài toán trên đoạn độ dài 0 . Kết hợp hai kết quả chia bài toán thì ta thấy bài toán trên đoạn độ dài $k+1$ được chia thành các bài toán con, mỗi bài toán trên đoạn độ dài 0. Nói cách khác, $A(k+1)$ đúng.

Tính dừng của thuật toán

Tính dừng

Thuật toán trên sẽ chia bài toán trên đoạn độ dài n thành các bài toán con, mỗi bài toán trên đoạn độ dài 0.

Gọi điều phải chứng minh là $A(n)$

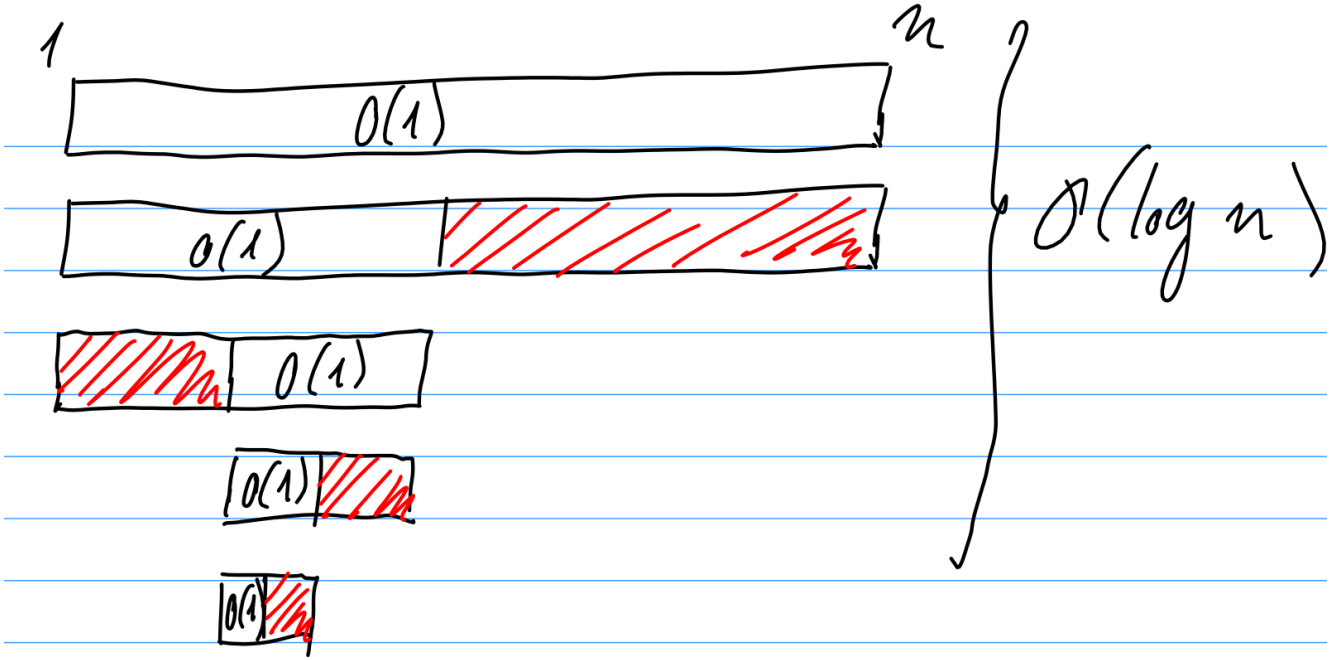
- $A(0)$ đúng.
- Giả sử $A(0), A(1), ..., A(k)$ đều đúng. Ta cần chứng minh $A(k+1)$ đúng.
  Thuật toán lúc đầu sẽ chia bài toán trên đoạn độ dài $k+1$ thành hai bài toán con với độ dài $\lfloor \frac{k+1}{2} \rfloor$ và $k - \lfloor \frac{k+1}{2} \rfloor$. Dễ thấy hai số này là các số nguyên không âm nhỏ hơn $k+1$, nên $A(\lfloor \frac{k+1}{2} \rfloor)$ và $A(k - \lfloor \frac{k+1}{2} \rfloor)$ đúng . Do đó ta sẽ có thể chia tiếp bài toán con trên đoạn độ dài $\lfloor \frac{k+1}{2} \rfloor$ và trên đoạn độ dài $k - \lfloor \frac{k+1}{2} \rfloor$ thành các bài toán con, mỗi bài toán trên đoạn độ dài 0 . Kết hợp hai kết quả chia bài toán thì ta thấy bài toán trên đoạn độ dài $k+1$ được chia thành các bài toán con, mỗi bài toán trên đoạn độ dài 0. Nói cách khác, $A(k+1)$ đúng.

Độ phức tạp tính toán

Gọi $T(n)$ là tổng số phép tính của thuật toán, ta có thể viết $T(n) = T(\frac{n}{2}) + O(1)$

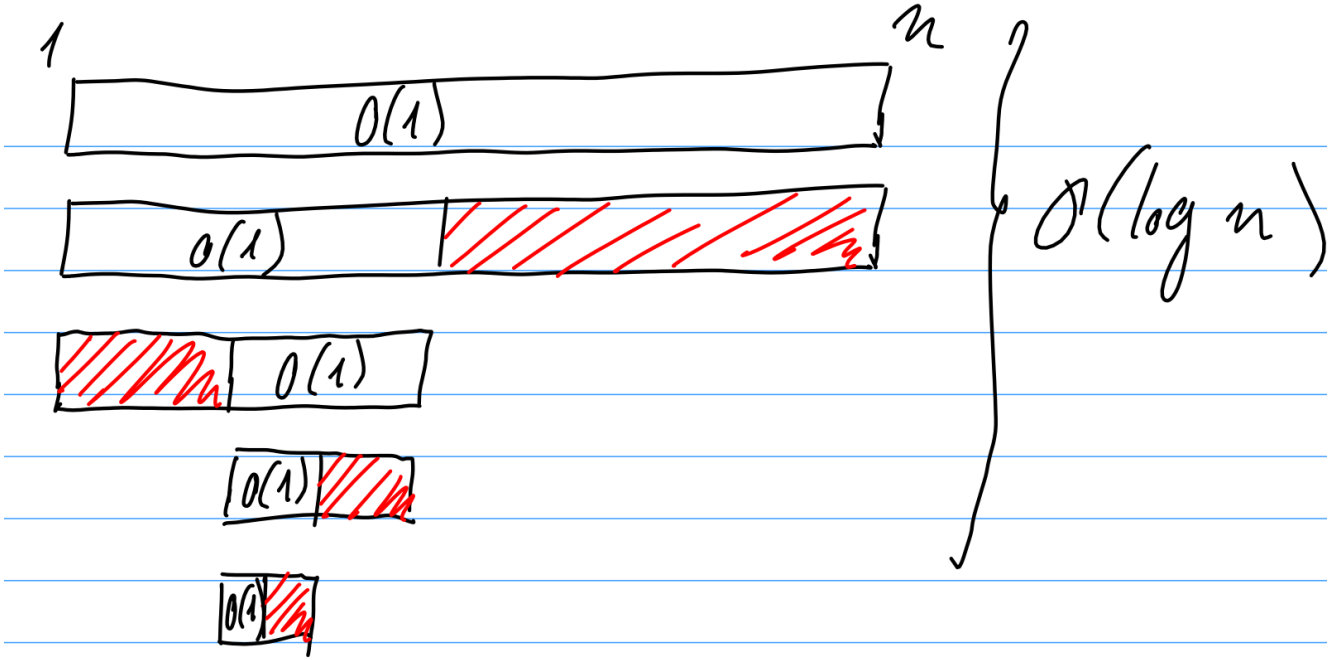
Độ phức tạp tính toán

Gọi $T(n)$ là tổng số phép tính của thuật toán, ta có thể viết $T(n) = T(\frac{n}{2}) + O(1)$



Độ phức tạp tính toán

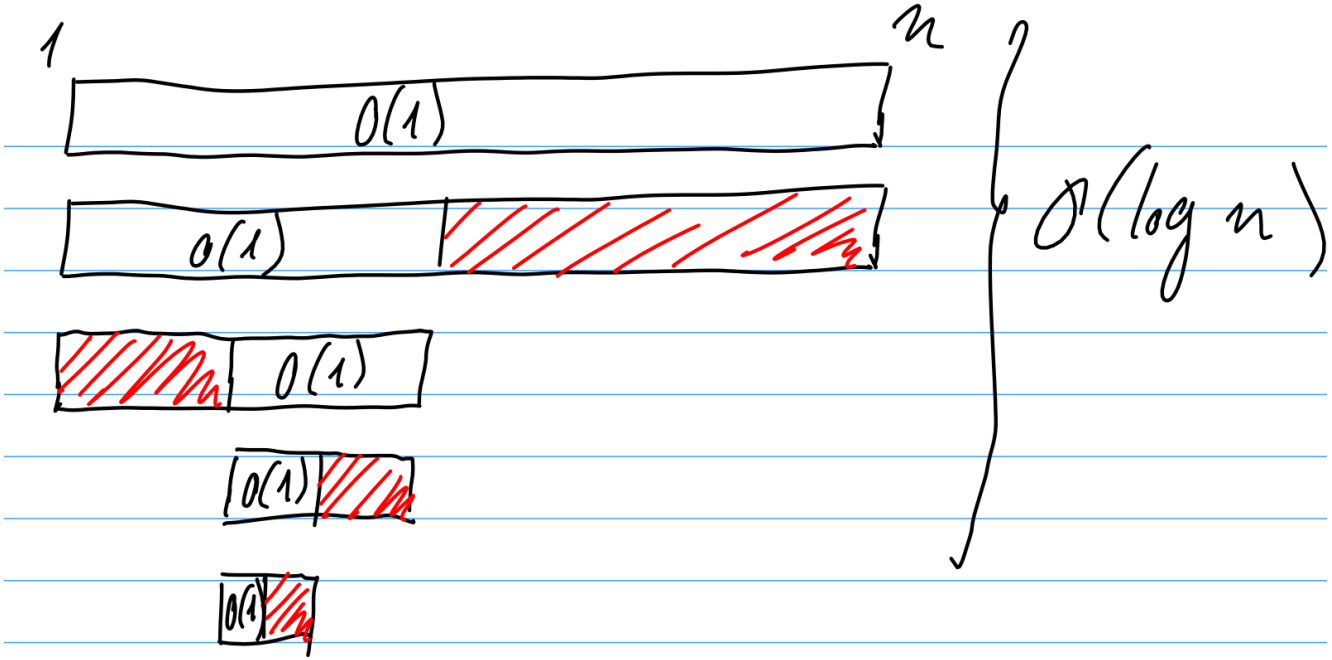
Gọi $T(n)$ là tổng số phép tính của thuật toán, ta có thể viết $T(n) = T(\frac{n}{2}) + O(1)$



Từ hình trên, theo quy tắc nhân, độ phức tạp tính toán của thuật toán là $O(\log n)$.

Độ phức tạp tính toán

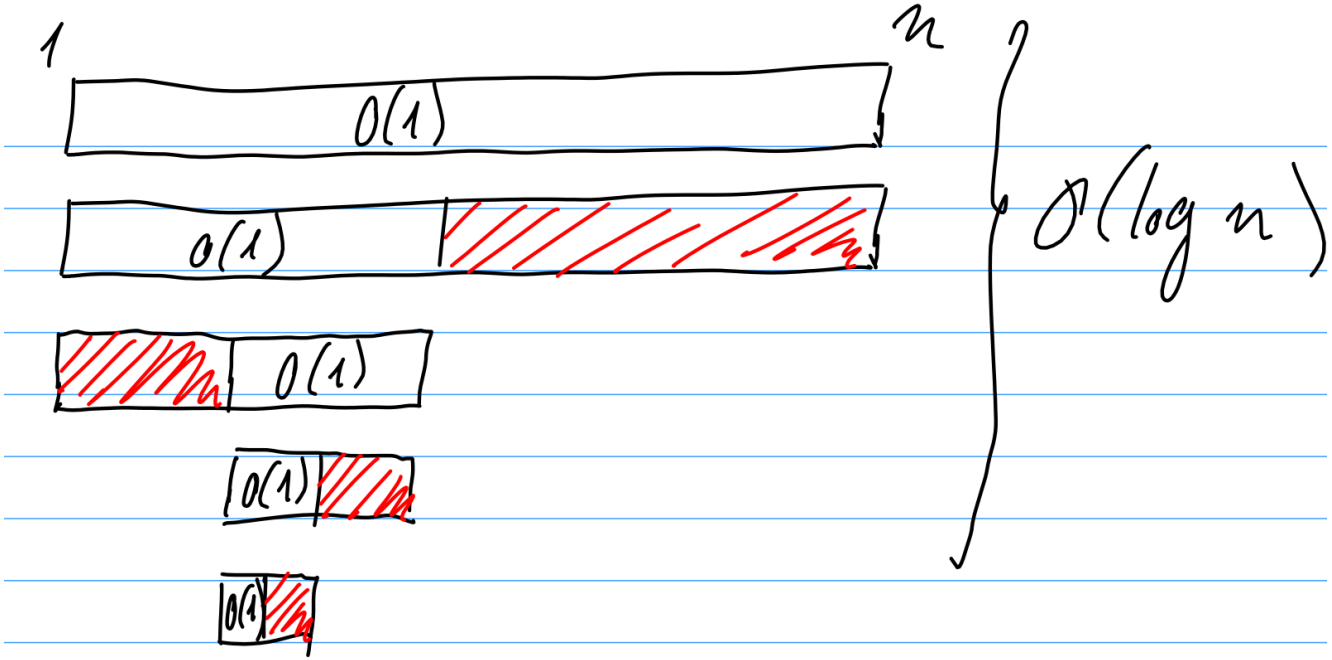
Gọi $T(n)$ là tổng số phép tính của thuật toán, ta có thể viết $T(n) = T(\frac{n}{2}) + O(1)$



Từ hình trên, theo quy tắc nhân, độ phức tạp tính toán của thuật toán là $O(\log n)$.
Với mọi $T(n) = T(\frac{n}{2}) + O(1)$, $T(n) \in O(\log n)$

Độ phức tạp tính toán

Gọi $T(n)$ là tổng số phép tính của thuật toán, ta có thể viết $T(n) = T(\frac{n}{2}) + O(1)$



Từ hình trên, theo quy tắc nhân, độ phức tạp tính toán của thuật toán là $O(\log n)$.
Với mọi $T(n) = T(\frac{n}{2}) + O(1)$, $T(n) \in O(\log n)$

Một số mẹo nên biết

Một số mẹo nên biết

- Kiểm tra sự tồn tại của một số k trong một dãy a tăng dần $\Rightarrow$ tìm số x lớn nhất nhỏ hơn hoặc bằng k, rồi so sánh x với k.

Một số mẹo nên biết

- Kiểm tra sự tồn tại của một số k trong một dãy a tăng dần $\Rightarrow$ tìm số x lớn nhất nhỏ hơn hoặc bằng k, rồi so sánh x với k.
- Nếu điều kiện tìm kiếm nhị phân được thỏa mãn, ta có thể tìm kiếm nhị phân và quy bài toán về kiểm tra xem một số nào đó có là nghiệm của bài toán hay không.

Một số mẹo nên biết

- Kiểm tra sự tồn tại của một số k trong một dãy a tăng dần $\Rightarrow$ tìm số x lớn nhất nhỏ hơn hoặc bằng k, rồi so sánh x với k.
- Nếu điều kiện tìm kiếm nhị phân được thỏa mãn, ta có thể tìm kiếm nhị phân và quy bài toán về kiểm tra xem một số nào đó có là nghiệm của bài toán hay không.
- Một số bài toán tiêu biểu:

Một số mẹo nên biết

- Kiểm tra sự tồn tại của một số k trong một dãy a tăng dần $\Rightarrow$ tìm số x lớn nhất nhỏ hơn hoặc bằng k, rồi so sánh x với k.
- Nếu điều kiện tìm kiếm nhị phân được thỏa mãn, ta có thể tìm kiếm nhị phân và quy bài toán về kiểm tra xem một số nào đó có là nghiệm của bài toán hay không.
- Một số bài toán tiêu biểu:
  - Bài 4 HSG Thành phố Hà Nội lớp 12 năm học 2015-2016.

Một số mẹo nên biết

- Kiểm tra sự tồn tại của một số k trong một dãy a tăng dần $\Rightarrow$ tìm số x lớn nhất nhỏ hơn hoặc bằng k, rồi so sánh x với k.
- Nếu điều kiện tìm kiếm nhị phân được thỏa mãn, ta có thể tìm kiếm nhị phân và quy bài toán về kiểm tra xem một số nào đó có là nghiệm của bài toán hay không.
- Một số bài toán tiêu biểu:
  - Bài 4 HSG Thành phố Hà Nội lớp 12 năm học 2015-2016.
  - "Chặt nhị phân" trên vnspoj.blogspot.com

Một số mẹo nên biết

- Kiểm tra sự tồn tại của một số k trong một dãy a tăng dần $\Rightarrow$ tìm số x lớn nhất nhỏ hơn hoặc bằng k, rồi so sánh x với k.
- Nếu điều kiện tìm kiếm nhị phân được thỏa mãn, ta có thể tìm kiếm nhị phân và quy bài toán về kiểm tra xem một số nào đó có là nghiệm của bài toán hay không.
- Một số bài toán tiêu biểu:
  - Bài 4 HSG Thành phố Hà Nội lớp 12 năm học 2015-2016.
  - "Chặt nhị phân" trên vnspoj.blogspot.com
  - oj.vnoi.info $\Rightarrow$ *TAG* $\Rightarrow$ Kĩ năng khác $\Rightarrow$ Tìm kiếm nhị phân.

Một số mẹo nên biết

- Kiểm tra sự tồn tại của một số k trong một dãy a tăng dần $\Rightarrow$ tìm số x lớn nhất nhỏ hơn hoặc bằng k, rồi so sánh x với k.
- Nếu điều kiện tìm kiếm nhị phân được thỏa mãn, ta có thể tìm kiếm nhị phân và quy bài toán về kiểm tra xem một số nào đó có là nghiệm của bài toán hay không.
- Một số bài toán tiêu biểu:
  - Bài 4 HSG Thành phố Hà Nội lớp 12 năm học 2015-2016.
  - "Chặt nhị phân" trên vnspoj.blogspot.com
  - oj.vnoi.info $\Rightarrow$ *TAG* $\Rightarrow$ Kĩ năng khác $\Rightarrow$ Tìm kiếm nhị phân.
  - Nhiều bài toán tương tác của `Codeforces`.

Tính x^n

Tính x^n

Bài toán

Cho số x và một số nguyên dương n, tính x^n trong $O(\log n)$

Tính x^n

Bài toán

Cho số x và một số nguyên dương n, tính x^n trong $O(\log n)$

Ý tưởng

- Dùng hàm `pow` trong `C++`?

Tính x^n

Bài toán

Cho số x và một số nguyên dương n, tính x^n trong $O(\log n)$

Ý tưởng

- Dùng hàm `pow` trong `C++`? $\leftarrow$ sai số.

Tính x^n

Bài toán

Cho số x và một số nguyên dương n, tính x^n trong $O(\log n)$

Ý tưởng

- Dùng hàm `pow` trong `C++`? $\leftarrow$ sai số.
- Dùng `**` trong `Python`?

Tính x^n

Bài toán

Cho số x và một số nguyên dương n, tính x^n trong $O(\log n)$

Ý tưởng

- Dùng hàm `pow` trong `C++`? $\leftarrow$ sai số.
- Dùng `**` trong `Python`? $\leftarrow$ độ phức tạp $O(n)$

Tính x^n

Bài toán

Cho số x và một số nguyên dương n, tính x^n trong $O(\log n)$

Ý tưởng

- Dùng hàm `pow` trong `C++`? $\leftarrow$ sai số.
- Dùng `**` trong `Python`? $\leftarrow$ độ phức tạp $O(n)$
- Tìm kiếm nhị phân kết quả?

Tính x^n

Bài toán

Cho số x và một số nguyên dương n, tính x^n trong $O(\log n)$

Ý tưởng

- Dùng hàm `pow` trong `C++`? $\leftarrow$ sai số.
- Dùng `**` trong `Python`? $\leftarrow$ độ phức tạp $O(n)$
- Tìm kiếm nhị phân kết quả? $\leftarrow$ không có cách kiểm tra kết quả nhanh chóng.

Chia để trị

Để tính x^n:

Chia để trị

Để tính x^n:

- Nếu $n = 0$ thì đáp án là 1.

Chia để trị

Để tính x^n:

- Nếu $n = 0$ thì đáp án là 1.
- Nếu $n > 0$ thì ta có $x^n = x^{\lfloor \frac{n}{2} \rfloor} x^{n - \lfloor \frac{n}{2} \rfloor}$

Chia để trị

Để tính x^n:

- Nếu $n = 0$ thì đáp án là 1.
- Nếu $n > 0$ thì ta có $x^n = x^{\lfloor \frac{n}{2} \rfloor} x^{n - \lfloor \frac{n}{2} \rfloor}$
  - Nếu n chẵn thì $\lfloor \frac{n}{2} \rfloor = n - \lfloor \frac{n}{2} \rfloor$, nên ta chỉ cần giải bài toán tính $x^{\lfloor \frac{n}{2} \rfloor}$ rồi tính $(x^{\lfloor \frac{n}{2} \rfloor})^2 = x^n$

Để tính x^n:

- Nếu $n = 0$ thì đáp án là 1.
- Nếu $n > 0$ thì ta có $x^n = x^{\lfloor \frac{n}{2} \rfloor} x^{n - \lfloor \frac{n}{2} \rfloor}$
  - Nếu n chẵn thì $\lfloor \frac{n}{2} \rfloor = n - \lfloor \frac{n}{2} \rfloor$, nên ta chỉ cần giải bài toán tính $x^{\lfloor \frac{n}{2} \rfloor}$ rồi tính $(x^{\lfloor \frac{n}{2} \rfloor})^2 = x^n$
  - Nếu n lẻ thì $n - \lfloor \frac{n}{2} \rfloor = \lfloor \frac{n}{2} \rfloor + 1$. Do đó $x^{n - \lfloor \frac{n}{2} \rfloor} = x^{\lfloor \frac{n}{2} \rfloor} \cdot x$.

Để tính x^n:

- Nếu $n = 0$ thì đáp án là 1.
- Nếu $n > 0$ thì ta có $x^n = x^{\lfloor \frac{n}{2} \rfloor} x^{n - \lfloor \frac{n}{2} \rfloor}$
  - Nếu n chẵn thì $\lfloor \frac{n}{2} \rfloor = n - \lfloor \frac{n}{2} \rfloor$, nên ta chỉ cần giải bài toán tính $x^{\lfloor \frac{n}{2} \rfloor}$ rồi tính $(x^{\lfloor \frac{n}{2} \rfloor})^2 = x^n$
  - Nếu n lẻ thì $n - \lfloor \frac{n}{2} \rfloor = \lfloor \frac{n}{2} \rfloor + 1$. Do đó $x^{n - \lfloor \frac{n}{2} \rfloor} = x^{\lfloor \frac{n}{2} \rfloor} \cdot x$. Vì vậy, ta chỉ cần tính $x^{\lfloor \frac{n}{2} \rfloor}$,

Chia để trị

Để tính x^n:

- Nếu $n = 0$ thì đáp án là 1.
- Nếu $n > 0$ thì ta có $x^n = x^{\lfloor \frac{n}{2} \rfloor} x^{n - \lfloor \frac{n}{2} \rfloor}$
  - Nếu n chẵn thì $\lfloor \frac{n}{2} \rfloor = n - \lfloor \frac{n}{2} \rfloor$, nên ta chỉ cần giải bài toán tính $x^{\lfloor \frac{n}{2} \rfloor}$ rồi tính $(x^{\lfloor \frac{n}{2} \rfloor})^2 = x^n$
  - Nếu n lẻ thì $n - \lfloor \frac{n}{2} \rfloor = \lfloor \frac{n}{2} \rfloor + 1$. Do đó $x^{n - \lfloor \frac{n}{2} \rfloor} = x^{\lfloor \frac{n}{2} \rfloor} \cdot x$. Vì vậy, ta chỉ cần tính $x^{\lfloor \frac{n}{2} \rfloor}$, sau đó tính $x^{\lfloor \frac{n}{2} \rfloor} x^{\lfloor \frac{n}{2} \rfloor} x$ là ra kết quả bài toán.

Để tính x^n:

- Nếu $n = 0$ thì đáp án là 1.
- Nếu $n > 0$ thì ta có $x^n = x^{\lfloor \frac{n}{2} \rfloor} x^{n - \lfloor \frac{n}{2} \rfloor}$
  - Nếu n chẵn thì $\lfloor \frac{n}{2} \rfloor = n - \lfloor \frac{n}{2} \rfloor$, nên ta chỉ cần giải bài toán tính $x^{\lfloor \frac{n}{2} \rfloor}$ rồi tính $(x^{\lfloor \frac{n}{2} \rfloor})^2 = x^n$
  - Nếu n lẻ thì $n - \lfloor \frac{n}{2} \rfloor = \lfloor \frac{n}{2} \rfloor + 1$. Do đó $x^{n - \lfloor \frac{n}{2} \rfloor} = x^{\lfloor \frac{n}{2} \rfloor} \cdot x$. Vì vậy, ta chỉ cần tính $x^{\lfloor \frac{n}{2} \rfloor}$, sau đó tính $x^{\lfloor \frac{n}{2} \rfloor} x^{\lfloor \frac{n}{2} \rfloor} x$ là ra kết quả bài toán.

  Tổng số phép tính để tính x^n là
  $T(n) = T(\frac{n}{2}) + O(1) \Rightarrow T(n) \in O(\log n)$

Sắp xếp dãy số

Sắp xếp dãy số

Sắp xếp

Cho một dãy a có n phần tử. Chỉ được phép so sánh các phần tử của dãy a, hãy sắp xếp dãy a tăng dần trong $O(n \log n)$

Sắp xếp dãy số

Sắp xếp

Cho một dãy a có n phần tử. Chỉ được phép so sánh các phần tử của dãy a, hãy sắp xếp dãy a tăng dần trong $O(n \log n)$

Ý tưởng (Chia để trị)

- Nếu $n = 1$ thì dãy số được cho đã được sắp xếp.

Sắp xếp dãy số

Sắp xếp

Cho một dãy a có n phần tử. Chỉ được phép so sánh các phần tử của dãy a, hãy sắp xếp dãy a tăng dần trong $O(n \log n)$

Ý tưởng (Chia để trị)

- Nếu $n = 1$ thì dãy số được cho đã được sắp xếp.
- Nếu $n > 1$ thì
  - Ta có thể chia dãy $a[1..n]$ thành hai dãy $a[1..\lfloor \frac{n}{2} \rfloor]$ và dãy $a[(\lfloor \frac{n}{2} \rfloor + 1)..n]$ trong $O(1)$

Sắp xếp dãy số

Sắp xếp

Cho một dãy a có n phần tử. Chỉ được phép so sánh các phần tử của dãy a, hãy sắp xếp dãy a tăng dần trong $O(n \log n)$

Ý tưởng (Chia để trị)

- Nếu $n = 1$ thì dãy số được cho đã được sắp xếp.
- Nếu $n > 1$ thì
  - Ta có thể chia dãy $a[1..n]$ thành hai dãy $a[1..\lfloor \frac{n}{2} \rfloor]$ và dãy $a[(\lfloor \frac{n}{2} \rfloor + 1)..n]$ trong $O(1)$
  - Sắp xếp hai dãy này.

Sắp xếp dãy số

Sắp xếp

Cho một dãy a có n phần tử. Chỉ được phép so sánh các phần tử của dãy a, hãy sắp xếp dãy a tăng dần trong $O(n \log n)$

Ý tưởng (Chia để trị)

- Nếu $n = 1$ thì dãy số được cho đã được sắp xếp.
- Nếu $n > 1$ thì
  - Ta có thể chia dãy $a[1..n]$ thành hai dãy $a[1..\lfloor \frac{n}{2} \rfloor]$ và dãy $a[(\lfloor \frac{n}{2} \rfloor + 1)..n]$ trong $O(1)$
  - Sắp xếp hai dãy này.
  - Kết hợp hai dãy đã được sắp xếp thành dãy $a[1..n]$ được sắp xếp với một thuật toán tối ưu nào đó.

Sắp xếp dãy số

Sắp xếp

Cho một dãy a có n phần tử. Chỉ được phép so sánh các phần tử của dãy a, hãy sắp xếp dãy a tăng dần trong $O(n \log n)$

Ý tưởng (Chia để trị)

- Nếu $n = 1$ thì dãy số được cho đã được sắp xếp.
- Nếu $n > 1$ thì
  - Ta có thể chia dãy $a[1..n]$ thành hai dãy $a[1..\lfloor \frac{n}{2} \rfloor]$ và dãy $a[(\lfloor \frac{n}{2} \rfloor + 1)..n]$ trong $O(1)$
  - Sắp xếp hai dãy này.
  - Kết hợp hai dãy đã được sắp xếp thành dãy $a[1..n]$ được sắp xếp với một thuật toán tối ưu nào đó.

Bài toán phụ

Cho hai dãy a và b đều là hai dãy tăng dần, hãy ghép hai dãy a và b thành một dãy c tăng dần trong $O(n)$

Bài toán phụ

Bài toán phụ

Cho hai dãy a và b đều là hai dãy tăng dần, hãy ghép hai dãy a và b thành một dãy c tăng dần trong $O(n)$

Ý tưởng

- Tạo hai con trỏ cho dãy a và dãy b. Ban đầu hai con trỏ này chỉ vào đầu mỗi dãy.

Bài toán phụ

Bài toán phụ

Cho hai dãy a và b đều là hai dãy tăng dần, hãy ghép hai dãy a và b thành một dãy c tăng dần trong $O(n)$

Ý tưởng

- Tạo hai con trỏ cho dãy a và dãy b. Ban đầu hai con trỏ này chỉ vào đầu mỗi dãy.
- Nếu như phần tử được chỉ ở dãy a nhỏ hơn phần tử được chỉ ở dãy b hoặc toàn bộ phần tử của dãy b đã được thêm vào c thì ta thêm phần tử ở dãy a và dịch con trỏ ở dãy a lên,

Bài toán phụ

Bài toán phụ

Cho hai dãy a và b đều là hai dãy tăng dần, hãy ghép hai dãy a và b thành một dãy c tăng dần trong $O(n)$

Ý tưởng

- Tạo hai con trỏ cho dãy a và dãy b. Ban đầu hai con trỏ này chỉ vào đầu mỗi dãy.
- Nếu như phần tử được chỉ ở dãy a nhỏ hơn phần tử được chỉ ở dãy b hoặc toàn bộ phần tử của dãy b đã được thêm vào c thì ta thêm phần tử ở dãy a và dịch con trỏ ở dãy a lên, và ngược lại.

Bài toán phụ

Bài toán phụ

Cho hai dãy a và b đều là hai dãy tăng dần, hãy ghép hai dãy a và b thành một dãy c tăng dần trong $O(n)$

Ý tưởng

- Tạo hai con trỏ cho dãy a và dãy b. Ban đầu hai con trỏ này chỉ vào đầu mỗi dãy.
- Nếu như phần tử được chỉ ở dãy a nhỏ hơn phần tử được chỉ ở dãy b hoặc toàn bộ phần tử của dãy b đã được thêm vào c thì ta thêm phần tử ở dãy a và dịch con trỏ ở dãy a lên, và ngược lại.

Dễ thấy phép tính tích cực của thuật toán là phép tính dịch con trỏ dãy $a \Rightarrow$ thuật toán có độ phức tạp $O(n)$

Bài toán phụ

Bài toán phụ

Cho hai dãy a và b đều là hai dãy tăng dần, hãy ghép hai dãy a và b thành một dãy c tăng dần trong $O(n)$

Ý tưởng

- Tạo hai con trỏ cho dãy a và dãy b. Ban đầu hai con trỏ này chỉ vào đầu mỗi dãy.
- Nếu như phần tử được chỉ ở dãy a nhỏ hơn phần tử được chỉ ở dãy b hoặc toàn bộ phần tử của dãy b đã được thêm vào c thì ta thêm phần tử ở dãy a và dịch con trỏ ở dãy a lên, và ngược lại.

Dễ thấy phép tính tích cực của thuật toán là phép tính dịch con trỏ dãy $a \Rightarrow$ thuật toán có độ phức tạp $O(n)$
Ta có thể chứng minh tính đúng đắn của thuật toán bằng phương pháp phản chứng (giả sử dãy c không phải dãy tăng dần, ...)

Bài toán phụ

Bài toán phụ

Cho hai dãy a và b đều là hai dãy tăng dần, hãy ghép hai dãy a và b thành một dãy c tăng dần trong $O(n)$

Ý tưởng

- Tạo hai con trỏ cho dãy a và dãy b. Ban đầu hai con trỏ này chỉ vào đầu mỗi dãy.
- Nếu như phần tử được chỉ ở dãy a nhỏ hơn phần tử được chỉ ở dãy b hoặc toàn bộ phần tử của dãy b đã được thêm vào c thì ta thêm phần tử ở dãy a và dịch con trỏ ở dãy a lên, và ngược lại.

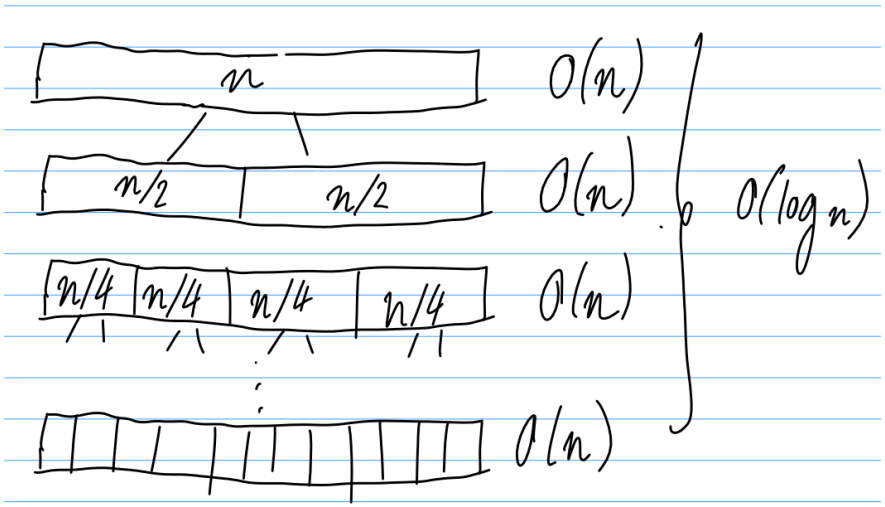
Dễ thấy phép tính tích cực của thuật toán là phép tính dịch con trỏ dãy $a \Rightarrow$ thuật toán có độ phức tạp $O(n)$
Ta có thể chứng minh tính đúng đắn của thuật toán bằng phương pháp phản chứng (giả sử dãy c không phải dãy tăng dần, ...)

Độ phức tạp tính toán

Gọi $T(n)$ là tổng số phép tính của thuật toán, ta có thể viết $T(n) = 2T(\frac{n}{2}) + O(n)$

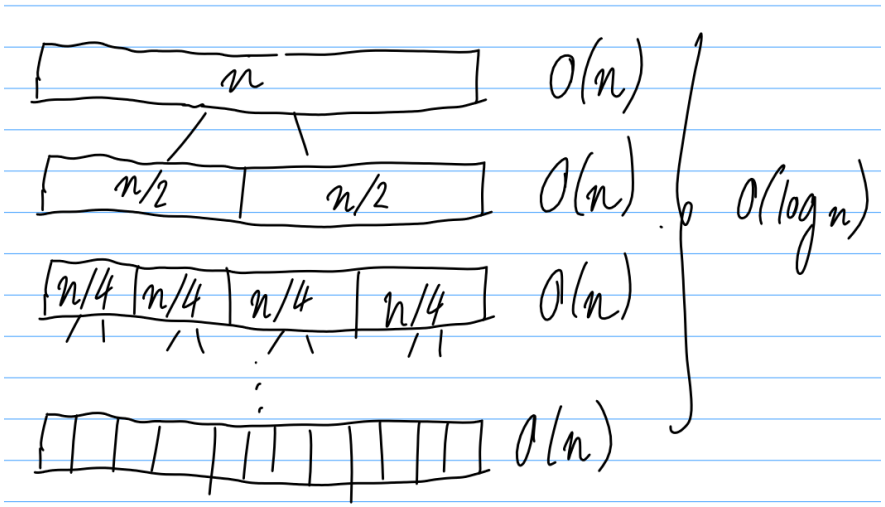
Độ phức tạp tính toán

Gọi $T(n)$ là tổng số phép tính của thuật toán, ta có thể viết $T(n) = 2T(\frac{n}{2}) + O(n)$



Độ phức tạp tính toán

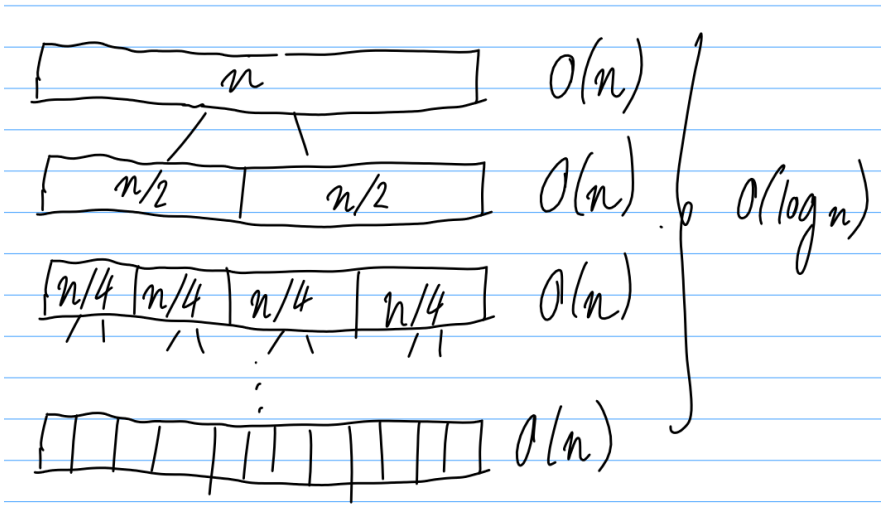
Gọi $T(n)$ là tổng số phép tính của thuật toán, ta có thể viết $T(n) = 2T(\frac{n}{2}) + O(n)$



Từ hình trên, theo quy tắc nhân, độ phức tạp tính toán của thuật toán là $O(n \log n)$.

Độ phức tạp tính toán

Gọi $T(n)$ là tổng số phép tính của thuật toán, ta có thể viết $T(n) = 2T(\frac{n}{2}) + O(n)$



Từ hình trên, theo quy tắc nhân, độ phức tạp tính toán của thuật toán là $O(n \log n)$.
Với mọi $T(n) = 2T(\frac{n}{2}) + O(n)$, $T(n) \in O(n \log n)$

Bài toán thứ 4

Bài toán thứ 4

Bài toán

Cho dãy số nguyên $a_1, a_2, ..., a_n$ $(2 \leq n \leq 10^5)$.
Tìm hai số nguyên i, j khác nhau sao cho $(i-j)^2 + (a_i - a_j)^2$ là nhỏ nhất.

Bài toán thứ 4

Bài toán

Cho dãy số nguyên $a_1, a_2, ..., a_n$ $(2 \leq n \leq 10^5)$.
Tìm hai số nguyên i, j khác nhau sao cho $(i-j)^2 + (a_i - a_j)^2$ là nhỏ nhất.

Ý tưởng

- $(a_i - a_j)^2 + (i-j)^2$ gợi cho ta nhớ đến công thức nào?

Bài toán thứ 4

Bài toán

Cho dãy số nguyên $a_1, a_2, ..., a_n$ $(2 \leq n \leq 10^5)$.
Tìm hai số nguyên i, j khác nhau sao cho $(i-j)^2 + (a_i - a_j)^2$ là nhỏ nhất.

Ý tưởng

- $(a_i - a_j)^2 + (i-j)^2$ gợi cho ta nhớ đến công thức nào?
- Làm thế nào để ứng dụng công thức đó vào bài toán này?

Bài toán thứ 4

Bài toán

Cho dãy số nguyên $a_1, a_2, ..., a_n$ $(2 \leq n \leq 10^5)$.
Tìm hai số nguyên i, j khác nhau sao cho $(i-j)^2 + (a_i - a_j)^2$ là nhỏ nhất.

Ý tưởng

- $(a_i - a_j)^2 + (i-j)^2$ gợi cho ta nhớ đến công thức nào?
- Làm thế nào để ứng dụng công thức đó vào bài toán này?
  - Ta sẽ biểu diễn số a_1 bằng điểm $(1, a_1)$, a_2 bằng điểm $(2, a_2)$, ... a_n bằng điểm (n, a_n).

Bài toán thứ 4

Bài toán

Cho dãy số nguyên $a_1, a_2, ..., a_n$ $(2 \leq n \leq 10^5)$.
Tìm hai số nguyên i, j khác nhau sao cho $(i-j)^2 + (a_i - a_j)^2$ là nhỏ nhất.

Ý tưởng

- $(a_i - a_j)^2 + (i-j)^2$ gợi cho ta nhớ đến công thức nào?
- Làm thế nào để ứng dụng công thức đó vào bài toán này?
  - Ta sẽ biểu diễn số a_1 bằng điểm $(1, a_1)$, a_2 bằng điểm $(2, a_2)$, ... a_n bằng điểm (n, a_n).
  - Khi đó, ta quy được bài toán về bài tìm cặp điểm gần nhau nhất trên mặt phẳng Oxy

Cặp điểm gần nhất

Bài toán

Cho n điểm $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ trên mặt phẳng Oxy. Tìm cặp điểm gần nhất.

Cặp điểm gần nhất

Bài toán

Cho n điểm $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ trên mặt phẳng Oxy. Tìm cặp điểm gần nhất.
Tạm thời, để cho đơn giản, ta giả sử không có hai điểm nào có cùng hoành độ.

Ý tưởng

- Nếu $n \leq 3$ thì ta dễ dàng giải được bài toán trong một vài phép tính bằng cách duyệt toàn bộ các cặp điểm.

Ý tưởng

- Nếu $n \leq 3$ thì ta dễ dàng giải được bài toán trong một vài phép tính bằng cách duyệt toàn bộ các cặp điểm.
- Nếu $n > 3$

Ý tưởng

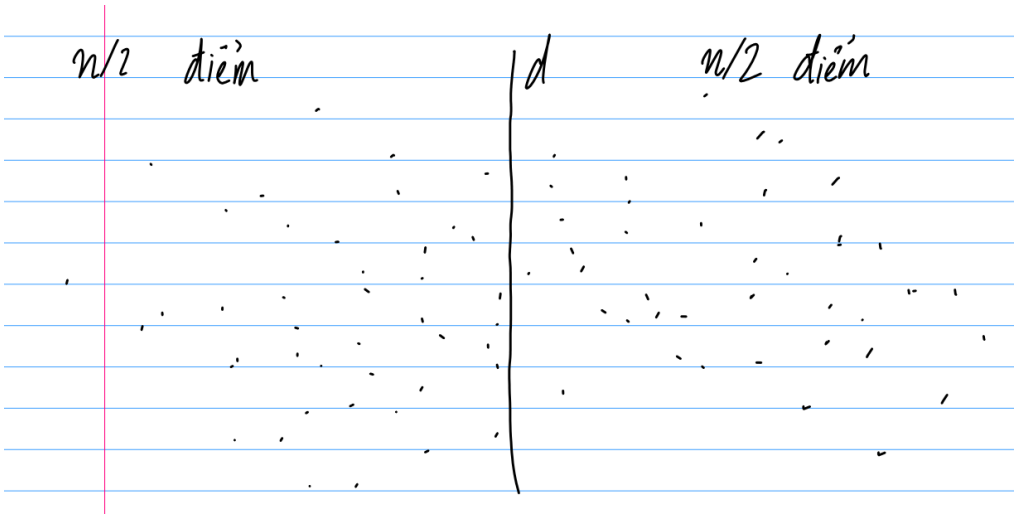
- Nếu $n \leq 3$ thì ta dễ dàng giải được bài toán trong một vài phép tính bằng cách duyệt toàn bộ các cặp điểm.
- Nếu $n > 3$
  - Ta có thể dùng một đường thẳng d để chia mặt phẳng thành hai nửa mặt phẳng sao cho mỗi nửa mặt phẳng có $\frac{n}{2}$ điểm.

Ý tưởng

- Nếu $n \leq 3$ thì ta dễ dàng giải được bài toán trong một vài phép tính bằng cách duyệt toàn bộ các cặp điểm.
- Nếu $n > 3$
  - Ta có thể dùng một đường thẳng d để chia mặt phẳng thành hai nửa mặt phẳng sao cho mỗi nửa mặt phẳng có $\frac{n}{2}$ điểm.
    Một cách chia: Sắp xếp n điểm theo hoành độ x tăng dần trong $O(n \log n)$, *lấy đường thẳng* $d : x = x_{\lfloor \frac{n+1}{2} \rfloor}$

Ý tưởng

- Nếu $n \leq 3$ thì ta dễ dàng giải được bài toán trong một vài phép tính bằng cách duyệt toàn bộ các cặp điểm.
- Nếu $n > 3$
  - Ta có thể dùng một đường thẳng d để chia mặt phẳng thành hai nửa mặt phẳng sao cho mỗi nửa mặt phẳng có $\frac{n}{2}$ điểm.
    Một cách chia: Sắp xếp n điểm theo hoành độ x tăng dần trong $O(n \log n)$, lấy đường thẳng $d : x = x_{\lfloor \frac{n+1}{2} \rfloor}$



Ý tưởng

- Cặp điểm gần nhất trong số n điểm là:

Ý tưởng

- Cặp điểm gần nhất trong số n điểm là:
  - Cặp điểm gần nhất ở nửa mặt phẳng bên trái (ta có thể gọi hàm đệ quy để giải) hoặc

Ý tưởng

- Cặp điểm gần nhất trong số n điểm là:
  - Cặp điểm gần nhất ở nửa mặt phẳng bên trái (ta có thể gọi hàm đệ quy để giải) hoặc
  - Cặp điểm gần nhất ở nửa mặt phẳng bên phải (ta có thể gọi hàm đệ quy để giải) hoặc

Ý tưởng

- Cặp điểm gần nhất trong số n điểm là:
  - Cặp điểm gần nhất ở nửa mặt phẳng bên trái (ta có thể gọi hàm đệ quy để giải) hoặc
  - Cặp điểm gần nhất ở nửa mặt phẳng bên phải (ta có thể gọi hàm đệ quy để giải) hoặc
  - Cặp điểm gần nhất trong số các cặp có một điểm ở bên trái và một điểm ở bên phải (ta cần tìm cách giải **nhanh** bài toán phụ này)

Ý tưởng

- Cặp điểm gần nhất trong số n điểm là:
  - Cặp điểm gần nhất ở nửa mặt phẳng bên trái (ta có thể gọi hàm đệ quy để giải) hoặc
  - Cặp điểm gần nhất ở nửa mặt phẳng bên phải (ta có thể gọi hàm đệ quy để giải) hoặc
  - Cặp điểm gần nhất trong số các cặp có một điểm ở bên trái và một điểm ở bên phải (ta cần tìm cách giải **nhanh** bài toán phụ này)

Bài toán phụ

Bài toán phụ

Ta có thể giải hai bài toán con trước để biết được kết quả của bài toán phụ chỉ ảnh hưởng đến kết quả của bài toán chính nếu cặp điểm của bài toán phụ có khoảng cách nhỏ hơn một số $\delta > 0$ nào đó.

Bài toán phụ

Ta có thể giải hai bài toán con trước để biết được kết quả của bài toán phụ chỉ ảnh hưởng đến kết quả của bài toán chính nếu cặp điểm của bài toán phụ có khoảng cách nhỏ hơn một số $\delta > 0$ nào đó.
Ta có thể làm như sau để giải bài toán phụ:

Bài toán phụ

Ta có thể giải hai bài toán con trước để biết được kết quả của bài toán phụ chỉ ảnh hưởng đến kết quả của bài toán chính nếu cặp điểm của bài toán phụ có khoảng cách nhỏ hơn một số $\delta > 0$ nào đó.

Ta có thể làm như sau để giải bài toán phụ:

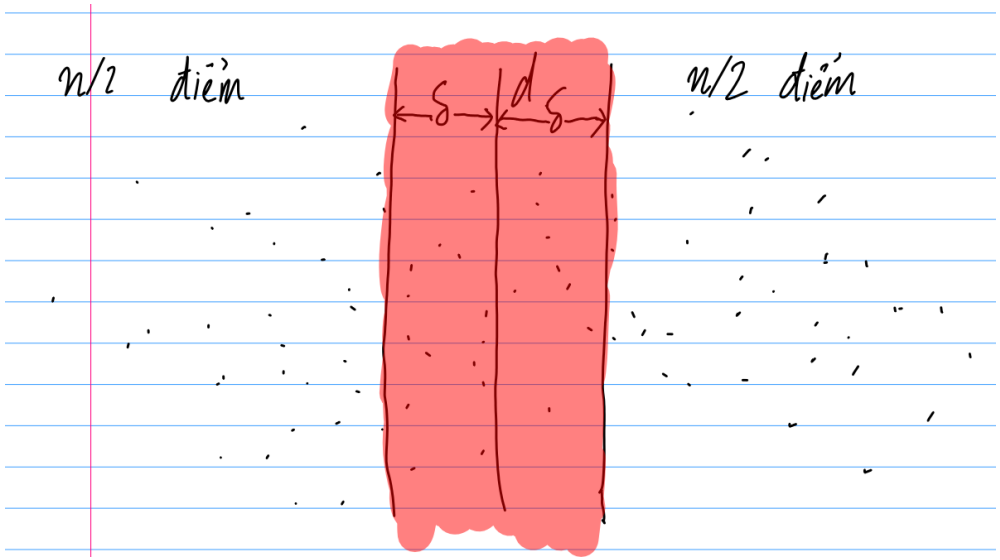
- Chọn các điểm có khoảng cách đến đường thẳng d nhỏ hơn δ. Ta chỉ cần xét các cặp điểm trong số các điểm này.

Bài toán phụ

Ta có thể giải hai bài toán con trước để biết được kết quả của bài toán phụ chỉ ảnh hưởng đến kết quả của bài toán chính nếu cặp điểm của bài toán phụ có khoảng cách nhỏ hơn một số $\delta > 0$ nào đó.

Ta có thể làm như sau để giải bài toán phụ:

- Chọn các điểm có khoảng cách đến đường thẳng d nhỏ hơn δ. Ta chỉ cần xét các cặp điểm trong số các điểm này.



Bài toán phụ

- Chọn các điểm có khoảng cách đến đường thẳng d nhỏ hơn δ. Ta chỉ cần xét các cặp điểm trong số các điểm này.
- Sắp xếp các điểm được chọn theo thứ tự tung độ tăng dần.

Bài toán phụ

- Chọn các điểm có khoảng cách đến đường thẳng d nhỏ hơn δ. Ta chỉ cần xét các cặp điểm trong số các điểm này.
- Sắp xếp các điểm được chọn theo thứ tự tung độ tăng dần.
- Trong dãy điểm được chọn đã được sắp xếp, với mỗi điểm, ta chỉ cần quan tâm tới các cặp điểm có chứa điểm đó và 7 điểm đứng sau đó trong dãy

Bài toán phụ

- Chọn các điểm có khoảng cách đến đường thẳng d nhỏ hơn δ. Ta chỉ cần xét các cặp điểm trong số các điểm này.
- Sắp xếp các điểm được chọn theo thứ tự tung độ tăng dần.
- Trong dãy điểm được chọn đã được sắp xếp, với mỗi điểm, ta chỉ cần quan tâm tới các cặp điểm có chứa điểm đó và 7 điểm đứng sau đó trong dãy

Chứng minh

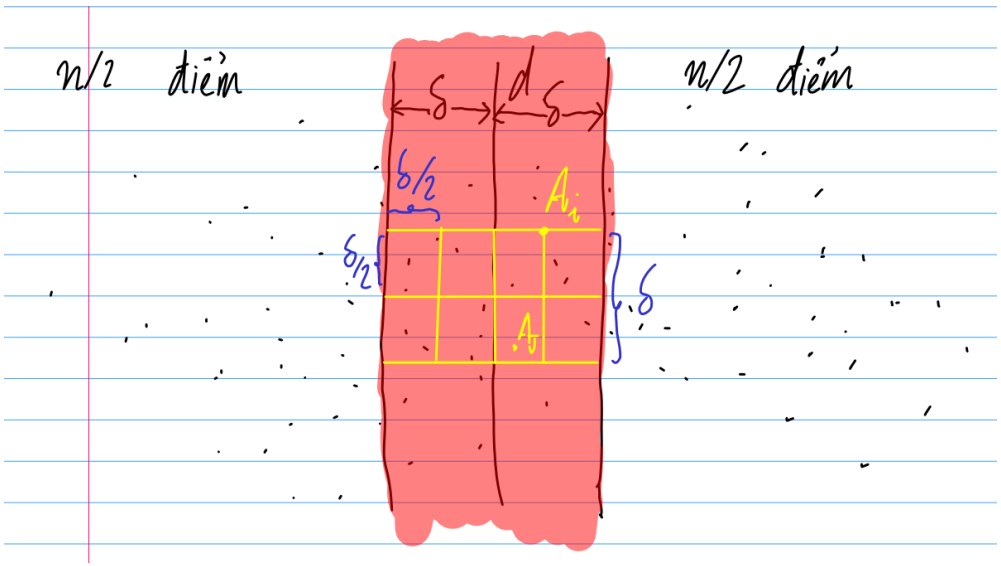
Gọi dãy điểm đã được sắp xếp là $A_1, A_2, A_3, ...$

Chứng minh

Gọi dãy điểm đã được sắp xếp là $A_1, A_2, A_3, ...$
Giả dụ tồn tại một cặp điểm cần được quan tâm mà ta chưa xét, tức hai điểm A_i và A_j có $j - i > 7$ mà có độ dài $A_iA_j < \delta$

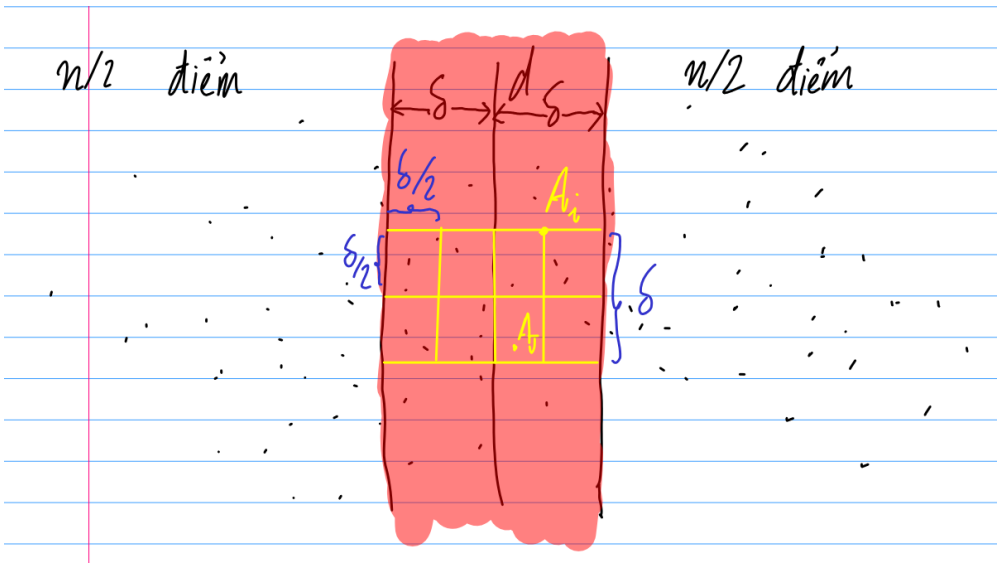
Chứng minh

Gọi dãy điểm đã được sắp xếp là $A_1, A_2, A_3, ...$
Giả dụ tồn tại một cặp điểm cần được quan tâm mà ta chưa xét, tức hai điểm A_i và A_j có $j - i > 7$ mà có độ dài $A_iA_j < \delta$
Vẽ 8 hình vuông cạnh $\frac{\delta}{2}$ như hình dưới đây.



Chứng minh

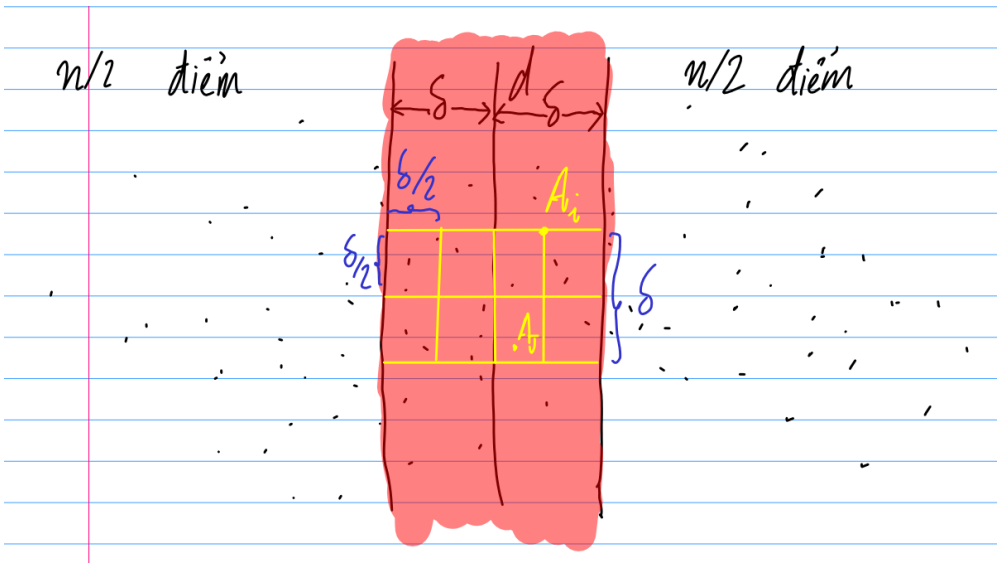
Gọi dãy điểm đã được sắp xếp là $A_1, A_2, A_3, ...$
Giả dụ tồn tại một cặp điểm cần được quan tâm mà ta chưa xét, tức hai điểm A_i và A_j có $j - i > 7$ mà có độ dài $A_iA_j < \delta$
Vẽ 8 hình vuông cạnh $\frac{\delta}{2}$ như hình dưới đây.



Do $A_iA_j < \delta$, dễ thấy các điểm $A_i, A_{i+1}, ..., A_j$ đều thuộc các hình vuông này.

Chứng minh

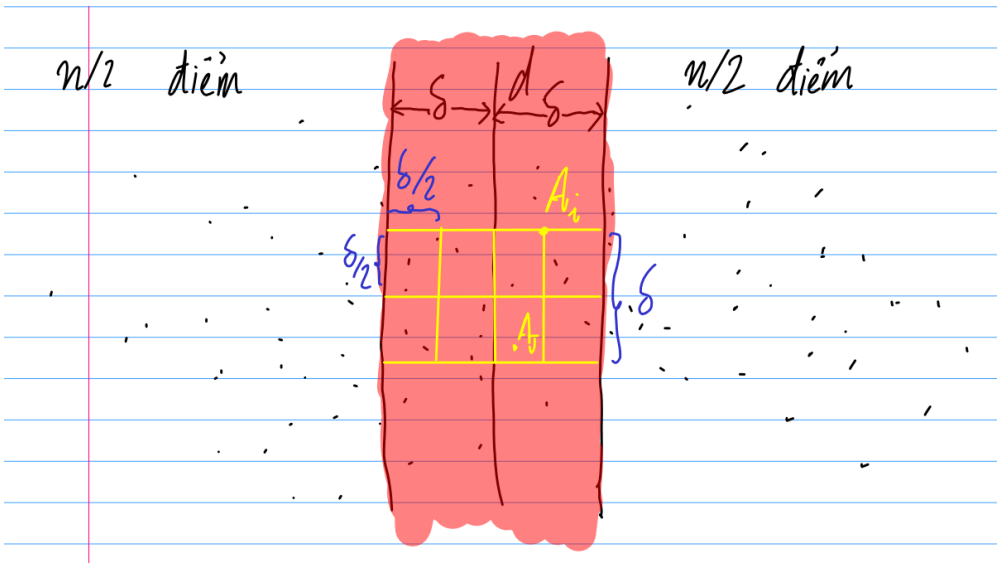
Gọi dãy điểm đã được sắp xếp là $A_1, A_2, A_3, ...$
Giả dụ tồn tại một cặp điểm cần được quan tâm mà ta chưa xét, tức hai điểm A_i và A_j có $j - i > 7$ mà có độ dài $A_iA_j < \delta$
Vẽ 8 hình vuông cạnh $\frac{\delta}{2}$ như hình dưới đây.



Do $A_iA_j < \delta$, dễ thấy các điểm $A_i, A_{i+1}, ..., A_j$ đều thuộc các hình vuông này. Do có 8 hình vuông; $A_i, A_{i+1}, ..., A_j$ là $j - i + 1$ điểm; và $j - i + 1 > 8$

Chứng minh

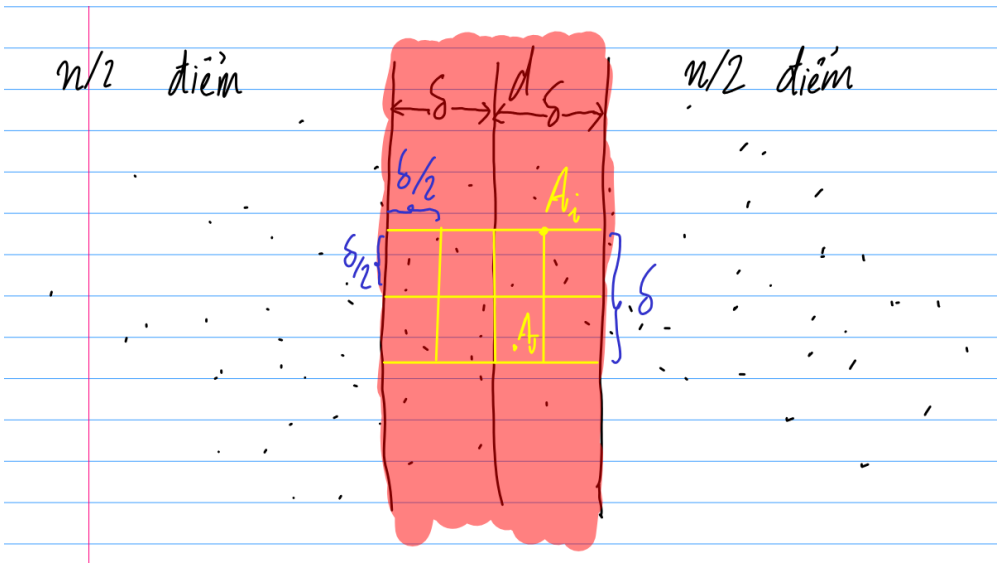
Gọi dãy điểm đã được sắp xếp là $A_1, A_2, A_3, ...$
Giả dụ tồn tại một cặp điểm cần được quan tâm mà ta chưa xét, tức hai điểm A_i và A_j có $j - i > 7$ mà có độ dài $A_iA_j < \delta$
Vẽ 8 hình vuông cạnh $\frac{\delta}{2}$ như hình dưới đây.



Do $A_iA_j < \delta$, dễ thấy các điểm $A_i, A_{i+1}, ..., A_j$ đều thuộc các hình vuông này. Do có 8 hình vuông; $A_i, A_{i+1}, ..., A_j$ là $j - i + 1$ điểm; và $j - i + 1 > 8$, theo nguyên lí nhốt thỏ, ta có thể chọn hai điểm A_x và A_y sao cho A_x và A_y cùng thuộc một hình vuông.

Chứng minh

Gọi dãy điểm đã được sắp xếp là $A_1, A_2, A_3, ...$
Giả dụ tồn tại một cặp điểm cần được quan tâm mà ta chưa xét, tức hai điểm A_i và A_j có $j - i > 7$ mà có độ dài $A_iA_j < \delta$
Vẽ 8 hình vuông cạnh $\frac{\delta}{2}$ như hình dưới đây.



Do $A_iA_j < \delta$, dễ thấy các điểm $A_i, A_{i+1}, ..., A_j$ đều thuộc các hình vuông này. Do có 8 hình vuông; $A_i, A_{i+1}, ..., A_j$ là $j - i + 1$ điểm; và $j - i + 1 > 8$, theo nguyên lí nhốt thỏ, ta có thể chọn hai điểm A_x và A_y sao cho A_x và A_y cùng thuộc một hình vuông.

Chứng minh

Do A_x và A_y cùng nằm trong một hình vuông cạnh $\frac{\delta}{2}$ nên:

Chứng minh

Do A_x và A_y cùng nằm trong một hình vuông cạnh $\frac{\delta}{2}$ nên:

- $A_xA_y \leq \frac{\delta\sqrt{2}}{2} < \delta$

Chứng minh

Do A_x và A_y cùng nằm trong một hình vuông cạnh $\frac{\delta}{2}$ nên:

- $A_x A_y \leq \frac{\delta\sqrt{2}}{2} < \delta$
- A_x và A_y cùng nằm trong một nửa mặt phẳng, nên $A_x A_y \geq \delta$

Chứng minh

Do A_x và A_y cùng nằm trong một hình vuông cạnh $\frac{\delta}{2}$ nên:

- $A_xA_y \leq \frac{\delta\sqrt{2}}{2} < \delta$
- A_x và A_y cùng nằm trong một nửa mặt phẳng, nên $A_xA_y \geq \delta$

Do hai điều này không thể đồng thời đúng, giả dụ ban đầu sai. Nói cách khác, mọi cặp điểm ta cần quan tâm đều đã được xét.

Độ phức tạp tính toán

Ở hàm đệ quy giải bài toán có n điểm, ta cần:

Độ phức tạp tính toán

Ở hàm đệ quy giải bài toán có n điểm, ta cần:

- Chia mặt phẳng thành hai phần, mỗi phần có $\frac{n}{2}$ điểm $\rightarrow$ cần $\boxed{O(n \log n)}$

Độ phức tạp tính toán

Ở hàm đệ quy giải bài toán có n điểm, ta cần:

- Chia mặt phẳng thành hai phần, mỗi phần có $\frac{n}{2}$ điểm $\rightarrow$ cần $\boxed{O(n \log n)}$
- Giải hai bài toán con $\rightarrow$ cần $2T(\frac{n}{2})$

Độ phức tạp tính toán

Ở hàm đệ quy giải bài toán có n điểm, ta cần:

- Chia mặt phẳng thành hai phần, mỗi phần có $\frac{n}{2}$ điểm $\rightarrow$ cần $\boxed{O(n \log n)}$
- Giải hai bài toán con $\rightarrow$ cần $2T(\frac{n}{2})$
- Chọn các điểm cách đường thẳng một khoảng $\delta \rightarrow$ cần $O(n)$

Độ phức tạp tính toán

Ở hàm đệ quy giải bài toán có n điểm, ta cần:

- Chia mặt phẳng thành hai phần, mỗi phần có $\frac{n}{2}$ điểm $\rightarrow$ cần $\boxed{O(n \log n)}$
- Giải hai bài toán con $\rightarrow$ cần $2T(\frac{n}{2})$
- Chọn các điểm cách đường thẳng một khoảng $\delta \rightarrow$ cần $O(n)$
- Sắp xếp các điểm được chọn $\rightarrow$ cần $\boxed{O(n \log n)}$

Độ phức tạp tính toán

Ở hàm đệ quy giải bài toán có n điểm, ta cần:

- Chia mặt phẳng thành hai phần, mỗi phần có $\frac{n}{2}$ điểm $\rightarrow$ cần $\boxed{O(n \log n)}$
- Giải hai bài toán con $\rightarrow$ cần $2T(\frac{n}{2})$
- Chọn các điểm cách đường thẳng một khoảng $\delta \rightarrow$ cần $O(n)$
- Sắp xếp các điểm được chọn $\rightarrow$ cần $\boxed{O(n \log n)}$
- Xét các cặp điểm để giải bài toán phụ $\rightarrow$ cần $O(n)$

Độ phức tạp tính toán

Ở hàm đệ quy giải bài toán có n điểm, ta cần:

- Chia mặt phẳng thành hai phần, mỗi phần có $\frac{n}{2}$ điểm $\rightarrow$ cần $\boxed{O(n \log n)}$
- Giải hai bài toán con $\rightarrow$ cần $2T(\frac{n}{2})$
- Chọn các điểm cách đường thẳng một khoảng $\delta \rightarrow$ cần $O(n)$
- Sắp xếp các điểm được chọn $\rightarrow$ cần $\boxed{O(n \log n)}$
- Xét các cặp điểm để giải bài toán phụ $\rightarrow$ cần $O(n)$

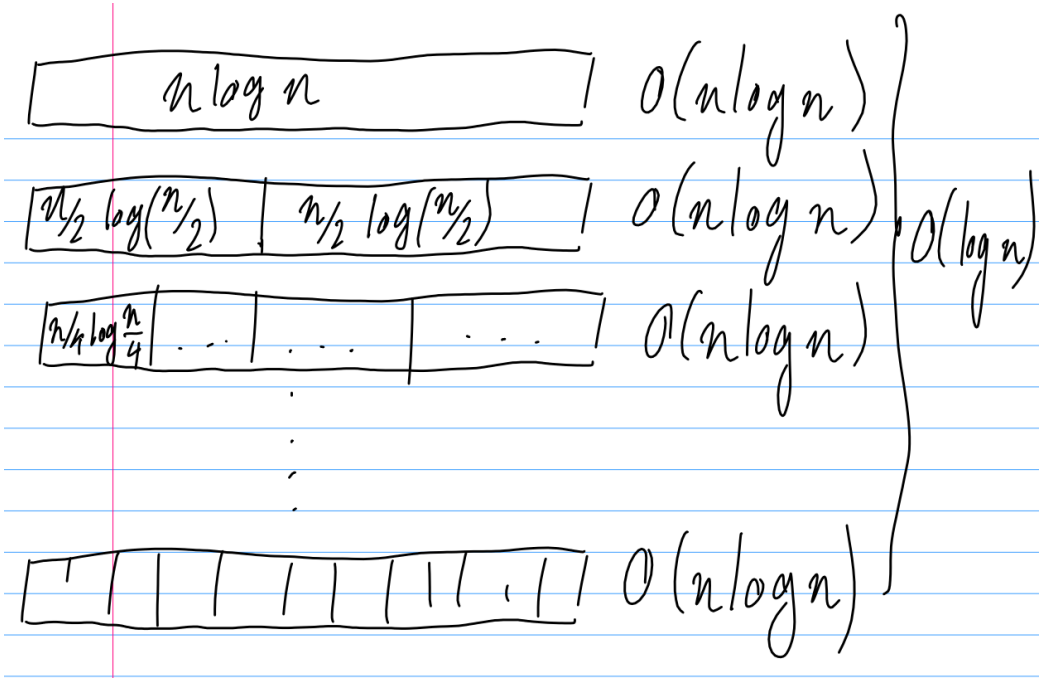
Vậy $T(n) = 2T(\frac{n}{2}) + O(n \log n)$

Độ phức tạp tính toán

$$T(n) = 2T(\tfrac{n}{2}) + O(n \log n)$$

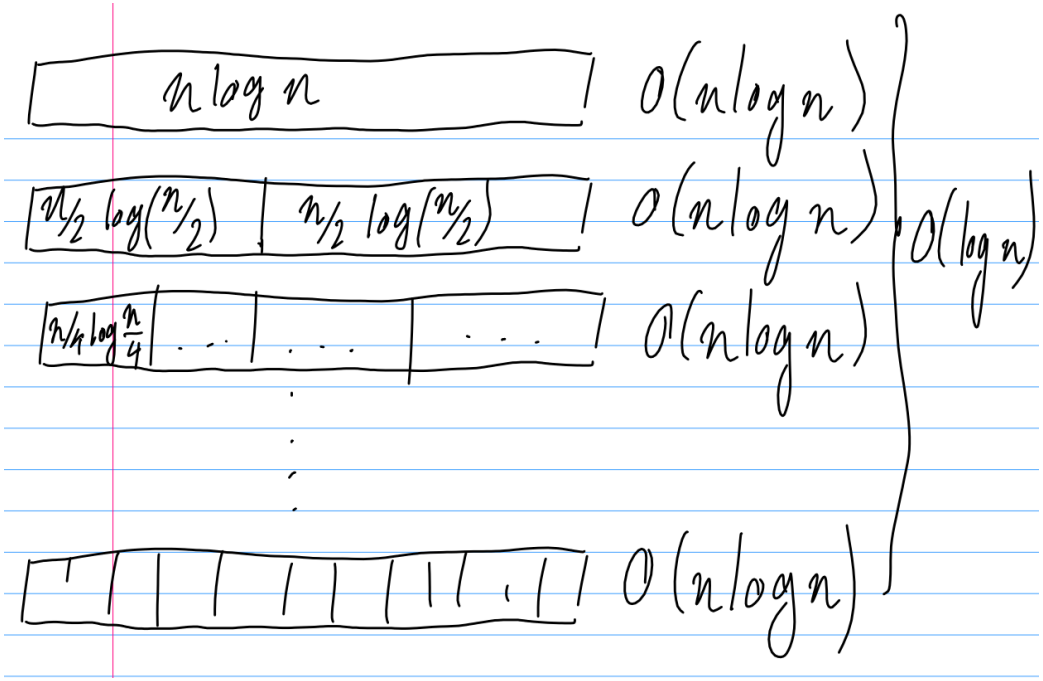
Độ phức tạp tính toán

$$T(n) = 2T(\tfrac{n}{2}) + O(n \log n)$$



Độ phức tạp tính toán

$$T(n) = 2T(\tfrac{n}{2}) + O(n \log n)$$



Từ hình trên, theo quy tắc nhân, độ phức tạp tính toán của thuật toán là $O(n \log^2 n)$.

Tổng kết

Để giải một bài toán theo phương pháp Chia để trị, ta cần:

Tổng kết

Để giải một bài toán theo phương pháp Chia để trị, ta cần:

1. Tìm cách giải nhanh bài toán nhỏ trong (tốt nhất là trong $O(1)$).

Tổng kết

Để giải một bài toán theo phương pháp Chia để trị, ta cần:

1. Tìm cách giải nhanh bài toán nhỏ trong (tốt nhất là trong $O(1)$).
2. Chia bài toán lớn thành các bài toán có kích thước nhỏ hơn (thường là hai phần tương đối bằng nhau)

Tổng kết

Để giải một bài toán theo phương pháp Chia để trị, ta cần:

1. Tìm cách giải nhanh bài toán nhỏ trong (tốt nhất là trong $O(1)$).
2. Chia bài toán lớn thành các bài toán có kích thước nhỏ hơn (thường là hai phần tương đối bằng nhau)
3. Xác định bài toán con nào ta thực sự cần giải (ví dụ: từ đáp án một bài toán con, ta có thể suy ra được đáp án bài toán con còn lại hay không), rồi gọi hàm đệ quy để giải các bài toán con đó.

Tổng kết

Để giải một bài toán theo phương pháp Chia để trị, ta cần:

1. Tìm cách giải nhanh bài toán nhỏ trong (tốt nhất là trong $O(1)$).
2. Chia bài toán lớn thành các bài toán có kích thước nhỏ hơn (thường là hai phần tương đối bằng nhau)
3. Xác định bài toán con nào ta thực sự cần giải (ví dụ: từ đáp án một bài toán con, ta có thể suy ra được đáp án bài toán con còn lại hay không), rồi gọi hàm đệ quy để giải các bài toán con đó.
4. Giải bài toán phụ (các trường hợp mà các bài toán con chưa bao quát hết)

Tổng kết

Để giải một bài toán theo phương pháp Chia để trị, ta cần:

1. Tìm cách giải nhanh bài toán nhỏ trong (tốt nhất là trong $O(1)$).
2. Chia bài toán lớn thành các bài toán có kích thước nhỏ hơn (thường là hai phần tương đối bằng nhau)
3. Xác định bài toán con nào ta thực sự cần giải (ví dụ: từ đáp án một bài toán con, ta có thể suy ra được đáp án bài toán con còn lại hay không), rồi gọi hàm đệ quy để giải các bài toán con đó.
4. Giải bài toán phụ (các trường hợp mà các bài toán con chưa bao quát hết)
5. Ghép các đáp án vào nhau để ra đáp án bài toán lớn.

Tổng kết

Để giải một bài toán theo phương pháp Chia để trị, ta cần:

1. Tìm cách giải nhanh bài toán nhỏ trong (tốt nhất là trong $O(1)$).
2. Chia bài toán lớn thành các bài toán có kích thước nhỏ hơn (thường là hai phần tương đối bằng nhau)
3. Xác định bài toán con nào ta thực sự cần giải (ví dụ: từ đáp án một bài toán con, ta có thể suy ra được đáp án bài toán con còn lại hay không), rồi gọi hàm đệ quy để giải các bài toán con đó.
4. Giải bài toán phụ (các trường hợp mà các bài toán con chưa bao quát hết)
5. Ghép các đáp án vào nhau để ra đáp án bài toán lớn.

Các bước 2, 3, 4, 5 cần được làm trong thời gian dưới $O(n \log n)$ để đảm bảo độ phức tạp tính toán cuối cùng dưới $O(n \log^2 n)$