Algorithms and Complexity Calculation Part 1

Course on Data Structure and Algorithms Using Python

Ankush Saklecha • Lesson 11 • Dec 17, 2023

$f(n) = \log(\sqrt{n}) \Rightarrow \log(\sqrt{10000}) = \log 100 = \underline{\underline{2}}$

$g(n) = \underline{\underline{\sqrt{n}}} \Rightarrow \sqrt{10000} = \underline{\underline{100}}$

$h(n) = \underline{\underline{n}} \Rightarrow \underline{\underline{10000}}$

$$\boxed{f < g < h}$$

How to calculate growth rate of function.

[1] by putting the value of n (larger value).

[2] if Exponential function the take the log of functions to reduce the growth and compare the resultant function.

unacademy

$$2^{n^2} > n^n$$

$n = 10$

$n = 10^{10}$

$\boxed{2^{100}}$ $\boxed{10^{10}}$

$\log(2^{n^2})$ $\log(n^n)$

$n^2 \log 2$ $n \log n$

$n * n * \log 2 > n * \log n$

$$2 \cdot 2^n \equiv 3 \cdot 2^n$$

$$2^n \simeq 8 \, 2^n$$

$$\boxed{4^n > 2^n}$$

$$\Downarrow \qquad\qquad \Downarrow$$

$$\log(4^n) \qquad \log(2^n)$$

$$n \log 4 > n \log 2$$

III By taking limit :→ $f(n) = n + 4$

$g(n) = n^2$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \frac{0}{1} = 0 \Rightarrow f(n) < g(n)$$

$$= \frac{1}{0} \infty \Rightarrow f(n) > g(n)$$

$$= \frac{1}{1} \text{ Const}^n \Rightarrow f(n) \cong g(n)$$

$(1, 2 \dots)$

$$\lim_{n\to\infty} \frac{n+4}{n^2} \Rightarrow \frac{\infty}{\infty}$$

$$\Rightarrow \lim_{n\to\infty} \frac{1}{2n} \Rightarrow \frac{1}{\infty} \Rightarrow \boxed{0} = \frac{0}{1}$$

$$n+4 < n^2$$

$f(n) = n^2 \log n$

$g(n) = n(\log n)^9$

$n \log n^9$

$9n \log n$

$\geq$

(a) $f(n) = O(g(n))$

~~(b)~~ $g(n) = O(f(n))$

(c) Both

(d) None

$n^2 \log n > n(\log n)^9$

$n \Rightarrow 10^{10}$

$(10^{10})^2 \log 10^{10}$ $\qquad$ $10^{10}(\log 10^{10})^9$

$10^{20} * 10$ $\qquad$ $10^{10}(10)^9$

$10^{21} > 10^{19}$

$n = 10^{100}$

$10^{200} * 100$ $\qquad$ $10^{100}(100)^9$

$10^{202} \gg 10^{118}$

Properties of Asymptotic Notation

① **Reflexive** :- aRa

[1] Big Oh ⇒

$$\left\{\begin{array}{l} f(n) = O(f(n)) \checkmark \\ f(n) = \Omega(f(n)) \checkmark \\ f(n) = \Theta(f(n)) \checkmark \end{array}\right\}$$

$f(n) = o(f(n))$ ✗

$f(n) = \omega(f(n))$ ✗

② Symmetric   aRb then bRa

[1] if $f(n) = O(g(n))$ then $g(n) = O(f(n))$

↳ not always true. So not Symmetric

② if $f(n) = \Omega(g(n))$ then $g(n) = \Omega(f(n))$

↳ not always true So not Symmetric

③ if $f(n) = \Theta(g(n))$ and $g(n) = \Theta(f(n))$

↳ it is Symmetric

(4) o

(5) ω $\Rightarrow$ No Never Symmetric

Asymmetric :- if aRb then $b \not R a$.

(1) $\emptyset \Rightarrow$ ✗

(2) $\Omega \Rightarrow$ ✗

(3) $\theta \Rightarrow$ ✗

(4) o $\Rightarrow$ ✓

(5) ω $\Rightarrow$ ✓

Antisymmetric :→

if (aRb and bRa) then $a = b$.

always true

1. O ✓
2. Ω ✓
3. Θ ✓
4. o ✓
5. ω ✓

if ($f(n) = o(g(n))$ and $g(n) = o(f(n))$) then $f(n) = g(n)$

⇓

always false

no need to check

a	b	$a \to b$
T	T	T
T	F	F
F	T	T
F	F	T

(if a then b)

satisfy

$a \Rightarrow$ ✓

$b \Rightarrow$ ✓

Transitive if $(aRb$ and $bRc)$ then aRc

1. O ✓ $a \leq b$ and $b \leq c$ then $a \leq c$
2. Ω ✓ if $a \geq b$ and $b \geq c$ then $a \geq c$
3. 8 ✓ if $a = b$ and $b = c$ then $a = c$
4. 0 ✓ if $a < b$ and $b < c$ then $a < c$
5. ω ✓ if $a > b$ and $b > c$ then $a > c$

Transpose :→

① if $f(n) = O(g(n))$ then $g(n) = \Omega(f(n))$

$O \longrightarrow \Omega$

$\Omega \longrightarrow O$

$\Theta \longrightarrow \Theta$

$o \longrightarrow \omega$

$\omega \longrightarrow o$

Asymptotic Complexity

1. $f(n) = (\log_{10}(n))^2$ and $g(n) = n$.

   (A) $f(n) = o(g(n))$ (B) $f(n) = \theta(g(n))$
   (C) $f(n) = \omega(g(n))$ (D) none of these

$f(n) = \log n * \log n < g(n) = \sqrt{n} * \sqrt{n}$

$2*2 \to 4$ $\Rightarrow 100$

2. Suppose $f(n) = \theta(n)$. Let $g(n) = f(n) + n$. Which is the strongest statement you can make?

   (A) $g(n) = O(n)$ (B) $g(n) = \Omega(n)$
   (C) $g(n) = \theta(n)$, i.e. both (a) and (b) are true (D) None of the above

$f(n) = \Theta(n) \Rightarrow n$

$g(n) = n+n \Rightarrow 2n$

$f(n) \simeq g(n)$

3. $f(n) = n^{2^n}$ and $g(n) = n^{2^n}$

   (A) $f(n) = o(g(n))$ × (B) $f(n) = \theta(g(n))$
   (C) $f(n) = \omega(g(n))$ × (D) none of these

$n \Rightarrow 10^{10}$ $f(n) = \frac{10^{10}}{10} \Rightarrow \boxed{10^9} > \log \log 10^{10}$

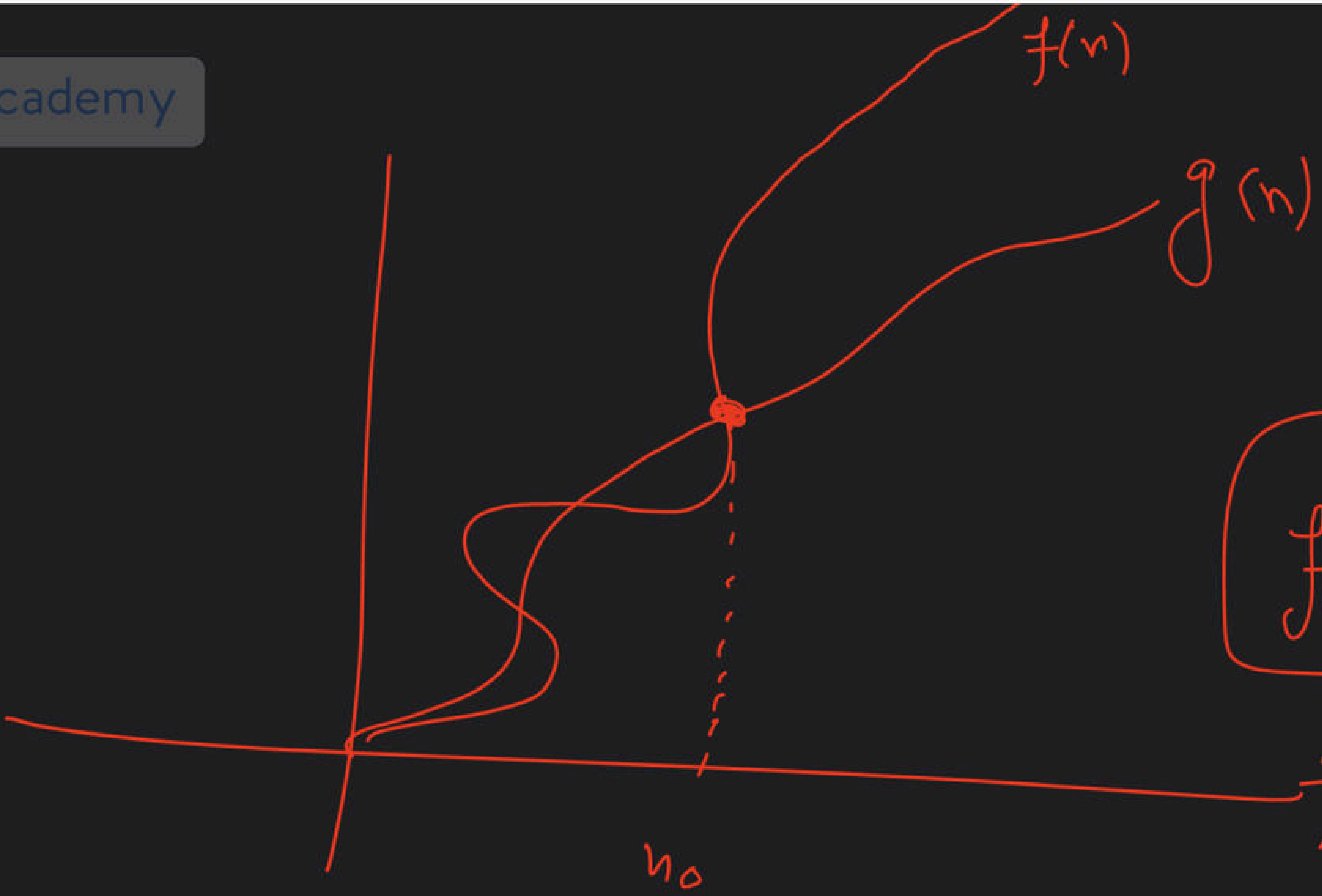
$\log 10 \Rightarrow \boxed{1}$

4. $f(n) = \frac{n}{\log n}$ $g(n) = \log\log(n)$

   (A) $f(n) = o(g(n))$ (B) $f(n) = \theta(g(n))$
   (C) $f(n) = \omega(g(n))$ (D) none of these

$n = 10^{100}$

$\boxed{f(n) > g(n)}$



$\boxed{f(n) = \Omega(g(n))}$

$f(n) = \omega(g(n))$

$g(n) = o(f(n))$

$g(n) = O(f(n))$

$f(n) = 100n + \log n \Rightarrow n + \log n \Rightarrow n$

$g(n) = n + (\log n)^2 \Rightarrow \rightarrow n$

5. f(n) = 100n + logn g(n) = n + $(\log n)^2$
   (A) f(n) = o(g(n))　　(B) f(n) = θ (g(n))
   (C) f(n) = ω (g(n))　　(D) none of these

1　　$1n^2 + n + 1$　　$2n^2 + 3n + 3$

f(n) ⇒ Equal bound

6. θ notation is more precise than Big-Oh notation at describing the growth of a function because:
   I. θ notation includes the ~~coefficient of the~~ highest order term. [F] [T]
   II. θ notation provides both an upper bound and a lower bound on the growth of a function. [T]
   III. Big-Oh notation suppresses lower order terms, so it does not necessarily accurately describe the behavior of a function for small values of N.
   (A) I only　　(B) I and II only
   (C) I and III only　　(D) I, II and III

$f(n) \le g(n)$ then $f(n) = O(g(n))$

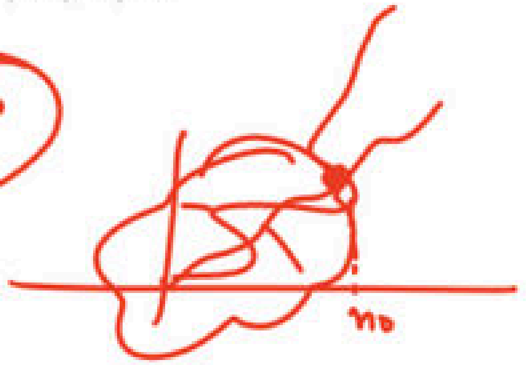
7. f(n) = n^2/logn g(n) = $n(\log n)^2$
   (A) f(n) = o(g(n))　　(B) f(n) = θ (g(n))
   (C) f(n) = ω (g(n))　　(D) none of these

$\boxed{f2,\ f1 = f4,\ f3}$

8. Arrange the following function in the ascending order of the growth rate
   f1(n) = nlogn　　f2(n) = $\sqrt{n}$　　f3(n) = n^3 + sin(n)　　f4(n) = $\log n^n$
   (A) f2, f4, f1, f3　　(B) f4, f1, f2, f3
   (C) f1, f4, f2, f3　　(D) Both B & C

↳ n logn

$n = 10^{10}$



$\dfrac{n^3 + 1}{n^3 - 1} \Rightarrow n^3$

$$f(n) = \frac{n^2}{\log n} \qquad g(n) = n(\log n)^2$$

$n = 10^{10}$

$$\frac{10^{20}}{10} \qquad 10^{10} * (10)^2$$

$$\Downarrow \qquad \qquad \Downarrow$$

$$10^{19} > 10^{12}$$

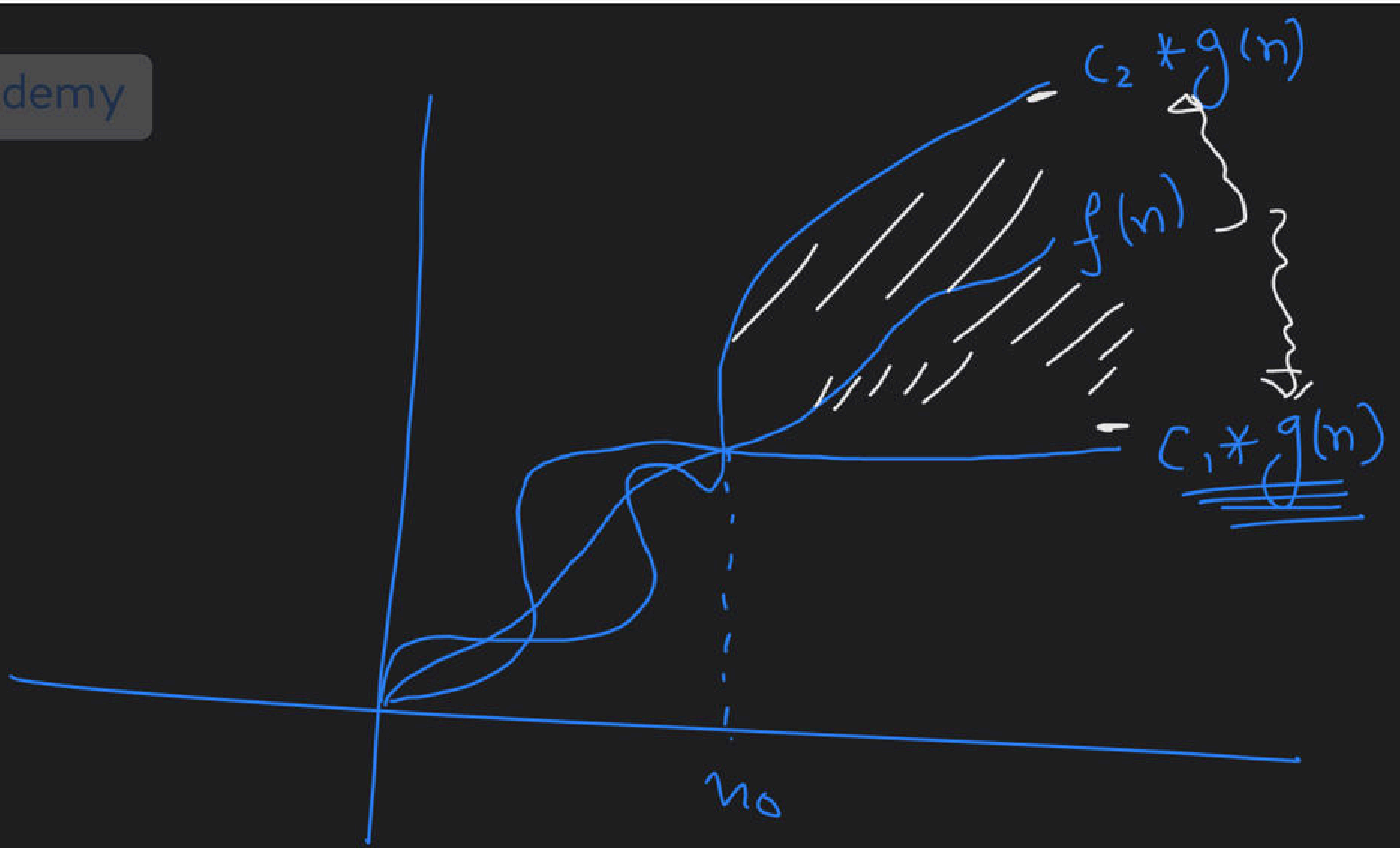
unacademy

$f(n) = n^2 + n + 2$

$f(n) = \Theta(n^2)$

$f(n) = O(n^3)$ true

$n^2 + n + 2 = O(n^3)$



$$f(n) = \Theta(g(n))$$

unacademy

$$f(n) = n^2 + n + 2$$

$$g(n) = n^2 + n$$

$$h(n) = n^2$$

Same

$$f(n) \equiv g(n) \equiv h(n)$$

$n = 10^{10} \Rightarrow [10]^{10} > \frac{10^{10}}{10} \Rightarrow 10^{9}$

9. f(n) = $(\log n)^{\log n}$ g(n) = n/log n

(A) f(n) = o(g(n)) (B) f(n) = θ (g(n))

(C) f(n) = ω (g(n)) (D) none of these

$n = 10^{100} \Rightarrow (100)^{100} \Rightarrow 10^{200} \gg \frac{10^{100}}{100} \Rightarrow 10^{18}$

10. f(n) = √n g(n) = $(\log n)^5$

(A) f(n) = o(g(n)) (B) f(n) = θ (g(n))

(C) f(n) = ω (g(n)) (D) none of these

$n = 10^{10} \Rightarrow \sqrt{10^{10}} \Rightarrow 10^{5} = 10^{5}$

$n = 10^{100} \Rightarrow \sqrt{10^{100}} \Rightarrow 10^{50} \gg (100)^{5} \Rightarrow 10^{10}$

11. Write the following function in the ascending order of the growth rate

f1(n) = $2^n + n^{100}$ f2(n) = 2^{2n} f3(n) = $\log_7 n$ f4(n) = $\sqrt{3n}$

f5(n) = $n^2 + \ln n$ f6(n) = 4 f7(n) = n

(A) f3,f4,f7,f5,f1,f2,f6 (B) f6,f3,f4,f7,f5,f1,f2

(C) f6,f3,f4,f7,f5,f2,f1 (D) f6,f4,f3,f7,f5,f1,f2

$f_1 \Rightarrow 2^n$

$f_2 \Rightarrow 4^n$

$f_3 = \log n$

$f_4 = \sqrt{n}$

$f_5 = n^2$

$f_6 = 1$

$f_7 = n$

$1, \log n, \sqrt{n}, n, n^2, 2^n, 4^n$

$f_6, f_3, f_4, f_7, f_5, f_1, f_2$

$(2^2)^n \Rightarrow 4^n$

Algorithm

[1] Sequential Algorithm $\Longrightarrow \underline{\underline{\Theta(1)}}$

(2) iterative Algorithm

(3) Recursive Algo.

I iterative Algorithm

```
for i in range(1, 10):
    Print i
```

10 times

$\Theta(1)$

```
for i in range(1, n):
    Print i
```

$\rightarrow$ 1 to n-1 $\rightarrow$ total $(n-1)$

$\Theta(n)$

```
for i in range(0, n):
    for j in range(0,n):
        Pass
```

$i = 0$ n

$i = 1$ n

$i =$ n

$n + n + \dots n$ times

$n * n$

$\Theta(n^2)$

$O(n^2)$

for i in range (0 to n):
    for j in range (0 to i):

$i=0$ 1
$i=1$ 2
$i=2$ 3
$i=n-1$ n

$1+2+3+\ldots\ldots n$ times

$$\frac{n(n+1)}{2} \Rightarrow \frac{n^2+n}{2}$$

$\Theta(n^2)$

```
for i in range(0, n):
    Print n

for j in range(0, n^2):
    Print j
```

$n + n^2$

$\Downarrow$

$\theta(n^2)$

Q for i in range $(1, n+1, i*2)$:
  Print i

$\hookrightarrow$ stepping

$1 + 1 + \cdots\cdots (k+1)$ times

$= k+1$

$\log_2 n + 1$

$\Rightarrow \Theta(\log n)$

$i = 1 = 2^0$ 1

$i = 2 = 2^1$ 1

$i = 4 = 2^2$ 1

$i = 8 = 2^3$ 1

$\vdots$

$i = 2^k \le n$ 1

$\boxed{2^k \le n}$ $\Rightarrow$ $k = \log_2 n$

```
for i in range(1, n+1, i*2):      -> log n
    for j in range(1, n+1, j*2):  -> log n
        - - - -
        - - -
        - - -
        - - - -
```

$\log n \times \log n$

$i = 1 \rightarrow \log n$

$i = 2 \rightarrow \log n$

$i = 2^2 \rightarrow \log n$

$\vdots$

$i = 2^K \leq n \rightarrow \log n$

$\hookrightarrow K = \log n$

$\log n + \log n + \dots (K+1)$

$= \log n * (K+1)$

$\Rightarrow \log n * (\log n + 1)$

$\Rightarrow (\log n)^2 + (\log n)$

$\Rightarrow \Theta((\log n)^2)$

```
for i in range(1, n+1, i*2):
    for j in range(1, i+1, j*2): ⇒ log i
```

$i = 1 \quad \log i \Rightarrow \log 1$

$i = 2 \quad \log i \Rightarrow \log 2$

$i = 2^2 \quad \log i \Rightarrow \log 2^2$

$\vdots \qquad \vdots$

$i = 2^k \le n \quad \log i \Rightarrow \log 2^k$

$\Rightarrow k = \log n$

$\log 1 + \log 2 + \log 2^2 + \cdots \log 2^k$

$0 + 1 + 2 + \cdots k$

$\Rightarrow \frac{k(k+1)}{2} \Rightarrow \frac{\log n(\log n+1)}{2}$

$\Theta((\log n)^2)$

for i in range(1, n+1, i*2): $\rightarrow \log n$

for j in range(1, n+1): $\rightarrow n$

$\times$

$\Theta(n \log n)$

unacademy

```
for i in range(1, n+1, i*2):
    for j in range(1, i+1):
        - - -
```

$i = 1$ 1

$i = 2$ 2

$i = 2^2$ 2^2

$\vdots$

$i = 2^k \le n$ 2^k

$$1 + 2 + 2^2 + \cdots \cdots 2^k$$

$$\Rightarrow \frac{1 \cdot (2^{k+1} - 1)}{(2-1)} \Rightarrow \frac{2 \cdot 2^k - 1}{(2-1)}$$

$$\Rightarrow 2 \cdot \underbrace{2^k}_{} - 1$$

$$\Rightarrow 2n - 1 \Rightarrow \boxed{\Theta(n)}$$

```
for i in range(1, n+1, i*5):
    for j in range(1, n+1, j*2):
        for k in range(1, n+1):
            - - - - -
```

$\log_5 n$ ×

$\log_2 n$ ×

n

$\Theta\left(n(\log n)^2\right)$

for i in range (1, n^2+1, i*2):

1 to n^2 $\quad$ i*2

$\log N \Rightarrow \log n^2 \Rightarrow 2\log n \Rightarrow \Theta(\log n)$

```
L1 for i in range(1, n+1, i*3):
L2     for j in range(1, n+1, j*3):
           Print(j)

L3     for k in range(1, n+1):
           Print(k)
```

$L_1 [L_2 + L_3]$

$\log_3 n [\log_3 n + n] \Rightarrow (\log n)^2 + n \log n$

$\longrightarrow \Theta(n \log n)$

```
i = 1
while (i<=n):
    Print i
    i = i + 1
```

$\rightarrow \theta(n)$

```
i = 1
while i <= n :
    Print i
    i = i * 2
```

$\Theta(\log n)$