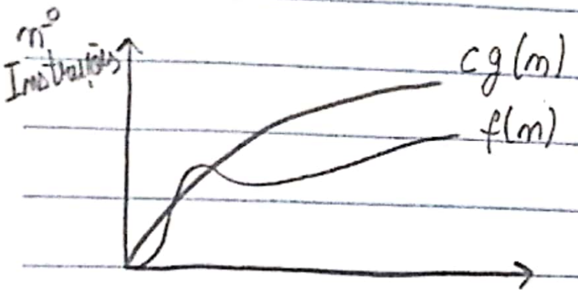
Aluno: Vinícius de Oliveira Guimarães e Vitor Enzo

① Big-O

É o limite superior do tempo de execução. Ou seja, é a abordagem pessimista de um algoritmo, particularmente adequada para o pior caso

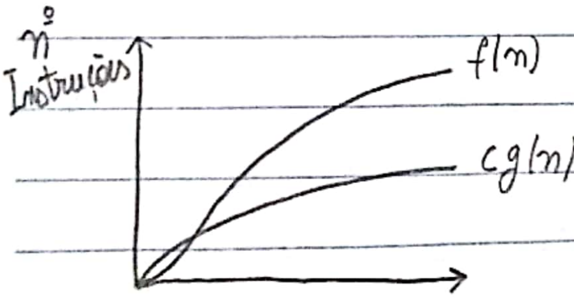
$f(n) = O(g(n))$ se existem constantes c e n_0 tais que $f(n) \leq c\,g(n)$ para todo $n \geq n_0$



② Notação Ω

Representa o limite inferior. Ou seja, é a abordagem mais otimista (Melhor caso).

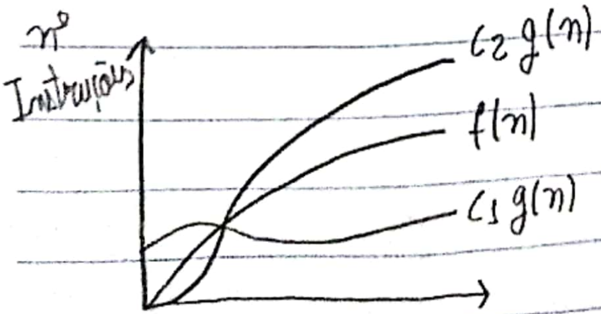
$f(n) = \Omega(g(n))$ se existem constantes c e n_0, tais que $f(n) \geq g(n)$ para todo $n \geq n_0$



③ Notação Θ

Serve para limitar função acima e abaixo simultaneamente (caso médio)

$f(n) = \Theta(g(n))$ se $\exists$ constantes $c_1, c_2 > 0$ e n_0 tais que $c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$



/ /

④ def Algo_c(n):

a = 100

j = n

while j > 0:

k = 0

$n.2 + (n-1).2 + (n-2).2 + \ldots$ { while k < j:

a = a + 10

k = k + 1

j = j - 1

return a

EXEMPLO (n = 3)

n = 3 j = 3 a = 100

j = 3

Antes	Antes	Depois	Depois
K=0	a=100	a=110	K=1
K=1	a=110	a=120	K=2
K=2	a=120	a=130	K=3

j = 2

Antes	Antes	Depois	Depois
K=0	a=130	a=140	K=1
K=1	a=140	a=150	K=2

$T(n) = 2 + \underbrace{1 + 1 + 1 + 1 + \ldots}_{n \text{ vezes}}$

$+ n.2 + (n-1).2 + (n-2).2 + (n-3).2 + \ldots + \underbrace{1 + 1 + 1 + 1 + 1}_{n \text{ vezes}}$

j = 1

Antes	Antes	Depois	Depois
K=0	a=150	a=160	K=1

j = 0

SAI DO WHILE EXTERNO

RETORNA a

Assim, $T(n) = 2 + n + \sum_{i=1}^{n} i.2 + n$

→ Sendo $\sum_{i=1}^{n} i.2 = n(n+1) = n^2 + n$, temos $T(n) = 2 + n + n^2 + n + n = 2 + 3n + n^2$

Como n^2 é o termo dominante, então o algoritmo é da ordem $\boxed{O(n^2)}$

⑤ for i in range(n)

fn1(i)

for j in range(n)

fn2(j)

for K in range(n)

fn3(j)

$\underbrace{n \cdot (1 + \overbrace{n \cdot (n + \overbrace{n \cdot (n^2)}^{\text{for } K})}^{\text{for } j})}_{\text{for } i}$

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

$n + n^3 + \underbrace{n^5}_{\substack{\text{Parte} \\ \text{Mais significativa} \\ \text{da expressão}}}$

⇒ Logo, sendo n^5 a parte dominante, temos que a eficiência do algoritmo é $\boxed{O(n^5)}$

7) $(n+a)^b$ é $\theta(n^b)$. Vamos analisar a expressão para diferentes valores de b

• $b=1$
$n+a$
Logo, $\theta(n)$

• $b=2$
$(n+a)^2 = n^2 + 2na + a^2$
Logo $\theta(n^2)$

• $b=3$
$(n+a)^3 = n^3 + a^3 + 3na(a+b)$
$= n^3 + a^3 + 3na^2 + 3nb^2$
Logo, $\theta(n^3)$

Ou seja, para todo valor de b, a expressão terá n^b como sendo o elemento dominante. Assim, $(n+a)^b$ é $\theta(n^b)$

8) $f(n) = n!$ é $O(n^n)$

$n! = n.(n-1).(n-2).(n-3)\dots 1 \leq \underbrace{n.n.n.n.n\dots n}_{n \text{ vezes o } n}$

Assim, como $n! \leq n^n$, então $n!$ é limitado superiormente por n^n. Ou seja, $\boxed{f(n) \text{ é } O(n^n)}$

9) $f(n) = \log n!$ é $O(n\log n)$

$\log n! = \log(2\pi n)^{1/2} + n\log\left(\frac{n}{e}\right) = \frac{1}{2}\log(2\pi n) + n(\log n - \log e)$

$= \underbrace{\frac{1}{2}\log(2\pi)}_{\text{CONSTANTE}} + \frac{1}{2}\log(n) + n\log n - \underbrace{n\log e}_{\text{CONSTANTE}}$

Como $n\log n$ domina $\frac{1}{2}\log(n)$, ou seja, $\frac{1}{2}\log(n)$ é limitado superiormente por $n\log n$, então $\boxed{f(n) \text{ é } O(n\log n)}$

10)
a) $f(n) = n^{1/2}$ e $g(n) = n^{2/3}$

$\frac{f(n)}{g(n)} = \frac{n^{1/2}}{n^{2/3}} = n^{1/2} \cdot n^{-2/3}$

$= n^{\left(\frac{1}{2} + \left(-\frac{2}{3}\right)\right)} = n^{-1/6} = \frac{1}{n^{1/6}}$

$\lim_{n\to\infty} \frac{1}{n^{1/6}} = \boxed{0}$, Logo, $g(n)$ é maior do que $f(n)$ e portanto, limita $f(n)$ superiormente

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

b) $f(n) = 10\log n$ e $g(n) = \log n^2$

$\lim_{n\to\infty} \frac{10\log n}{\log n^2} = 10 \cdot \lim_{n\to\infty} \frac{\cancel{\log n}}{2\cancel{\log n}}$

$= 10 \cdot \lim_{n\to\infty} \frac{1}{2} = \frac{10}{2} = \boxed{5}$

Como deu constante, então $f(n)$ e $g(n)$ crescem na mesma proporção. Logo,

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

⑥ $g(n) = \sum_{k=0}^{n} c^k$ é
a) $\theta(n)$, se $c=1$
b) $\theta(c^n)$, se $c>1$
c) $\theta(1)$, se $c<1$

a) $c=1$ $\sum_{k=0}^{n} c^k = 1^0 + \underbrace{1^1+1^2+1^3+1^4+1^5+\ldots+1^n}$

$1 + n.1 = \boxed{1+n}$

Sendo $g(n) = 1+n$, então $g(n)$ é da ordem $O(n)$, pois n é dominante na expressão

b) $c>1$ $\sum_{k=0}^{n} c^k = 1 + c^1 + c^2 + c^3 + c^5 + c^6 + \ldots + c^n$

Quando n vai para o infinito, c^n vai para o infinito também, sendo o elemento dominante no somatório. Assim, $g(n)$ é da ordem $\boxed{O(c^n)}$

c) $c<1$ $g(n) = \sum_{k=0}^{n} c^k = 1 + c^1 + c^2 + c^3 + c^4 + \ldots + c^n$

$S_{n+1} = a_1 + a_1 \cdot q + a_1 \cdot q^2 + a_1 \cdot q^3 + \ldots + a_1 \cdot q^n$
$qS_{n+1} = a_1 \cdot q + a_1 \cdot q^2 + a_1 \cdot q^3 + \ldots + a_1 \cdot q^{n+1}$

$S_{n+1} - qS_{n+1} = a_1 - a_1 \cdot q^{n+1}$
$S_{n+1}(1-q) = a_1 - a_1 \cdot q^{n+1}$

$$\boxed{S_{n+1} = \frac{a_1 - a_1 \cdot q^{n+1}}{1-q}} = \boxed{\frac{a_1 \cdot (1-q^{n+1})}{1-q}}$$

Assim, $g(n) = \dfrac{1 \cdot (1 - c^{n+1})}{1-c}$

Sendo $c<1$, quando n for para o infinito, c^{n+1} tenderá a zero. Logo, ficamos com $g(n) = \dfrac{1 . (1-0)}{1-c} = 1 . \dfrac{1}{1-c}$. Olhando para a expressão resultante de $g(n)$, sabemos que $g(n)$ é da ordem de $O(1)$, pois a nossa expressão é constante. Assim, $\boxed{g(n) \text{ é } O(1)}$

Portanto, conclui-se a) $c=1$ cresce linearmente enquanto que b) $c>1$ cresce exponencialmente e c) $c<1$ tem eficiência constante

C) $f(n) = \sqrt{n}$ e $g(n) = (\log n^3)$

$$\lim_{n\to\infty} \frac{n^{1/2}}{\log n^3} = \lim_{n\to\infty} \frac{n^{1/2}}{3\log n} = \frac{1}{3} \lim_{n\to\infty} \frac{(n^{1/2})'}{(\log n)'} = \frac{1}{3} \lim_{n\to\infty} \frac{\frac{1}{2} n^{-1/2}}{\frac{1}{n}}$$

$$= \frac{1}{3} \lim \frac{\frac{1}{2} \cdot \frac{1}{\sqrt{n}}}{\frac{1}{n}} = \frac{1}{3} \lim_{n\to\infty} \frac{n}{2\sqrt{n}} = \frac{1}{3} \lim_{n\to\infty} \frac{1}{2} \sqrt{n} = \boxed{\infty}$$

Como deu ∞, então $f(n)$ cresce mais rápido do que $g(n)$

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

D) $f(n) = n^{0,01}$
$g(n) = \log n$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{(n^{0,01})}{(\log n)}$$

Quando n é grande, temos que $g(n)$ é menor do que $f(n)$

$$= \lim_{n\to\infty} 0,01 \cdot n^{0,01} = \boxed{\infty}$$

Assim, pode-se dizer que $f(n)$ cresce mais rápido do que $g(n)$

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