NOÇÕES DE COMPLEXIDADE DE ALGORITMOS

PUC MINAS

ALGORITMOS E ESTRUTURAS DE DADOS II

EXERCÍCIO RESOLVIDO (1)

- Plote um gráfico com todas as funções abaixo:

a) $f(n) = n^3$

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

c) $f(n) = n * \log(n)$

d) $f(n) = n$

e) $f(n) = \text{sqrt}(n)$

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

EXERCÍCIO RESOLVIDO (1)

- Plote um gráfico com todas as funções abaixo:

a) $f(n) = n^3$

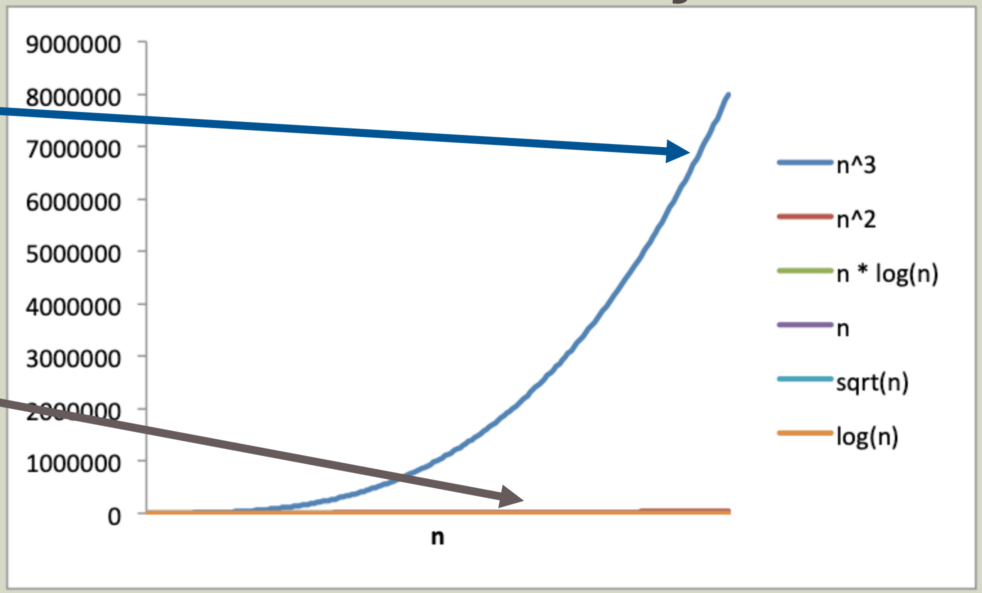
b) $f(n) = n^2$

c) $f(n) = n * \log(n)$

d) $f(n) = n$

e) $f(n) = \text{sqrt}(n)$

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



EXERCÍCIO RESOLVIDO (1)

- Plote um gráfico com todas as funções abaixo:

a) $f(n) = n^3$

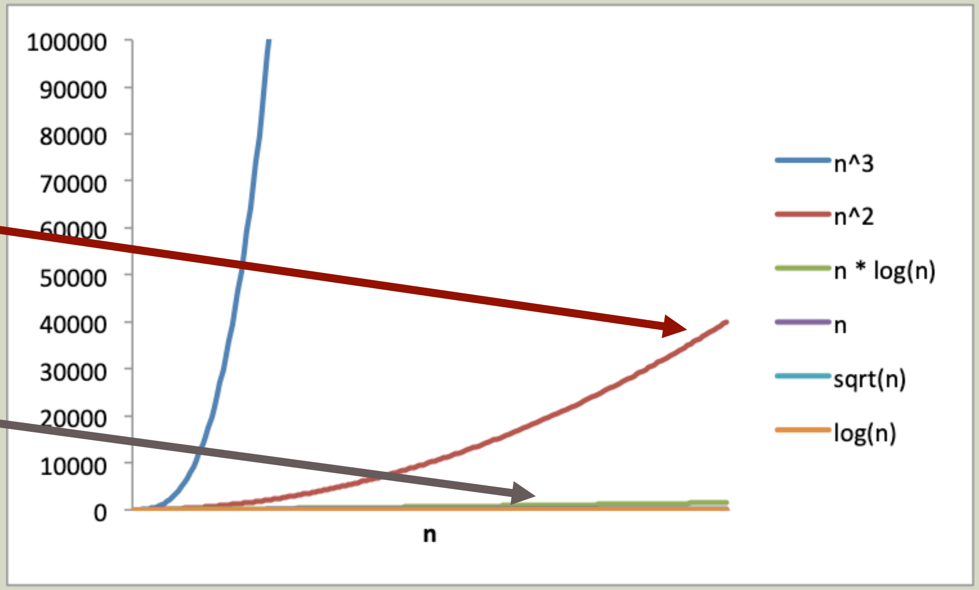
b) **$f(n) = n^2$**

c) $f(n) = n * \log(n)$

d) $f(n) = n$

e) $f(n) = \text{sqrt}(n)$

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



EXERCÍCIO RESOLVIDO (1)

- Plote um gráfico com todas as funções abaixo: ✓

a) $f(n) = n^3$

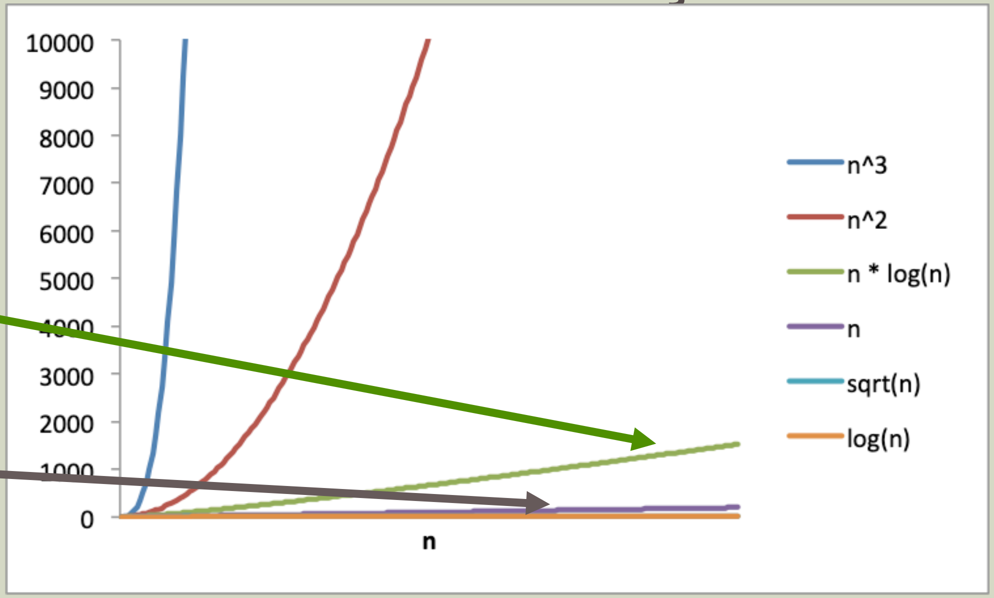
b) $f(n) = n^2$

c) **$f(n) = n * \log(n)$**

d) $f(n) = n$

e) $f(n) = \text{sqrt}(n)$

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



EXERCÍCIO RESOLVIDO (1)

- Plote um gráfico com todas as funções abaixo:

a) $f(n) = n^3$

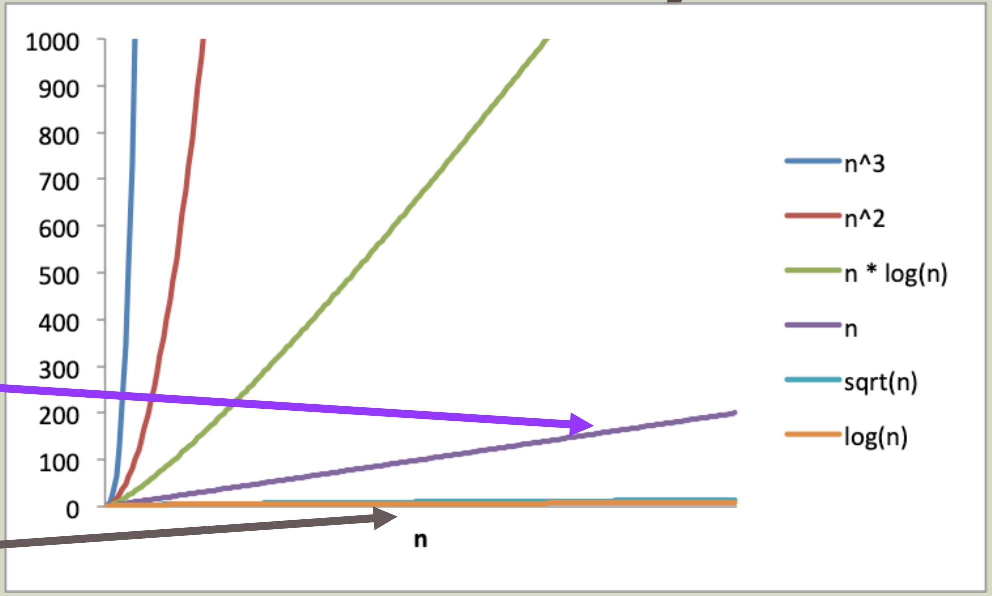
b) $f(n) = n^2$

c) $f(n) = n * \log(n)$

d) **$f(n) = n$**

e) $f(n) = \text{sqrt}(n)$

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



EXERCÍCIO RESOLVIDO (1)

- Plote um gráfico com todas as funções abaixo: ✓

a) $f(n) = n^3$

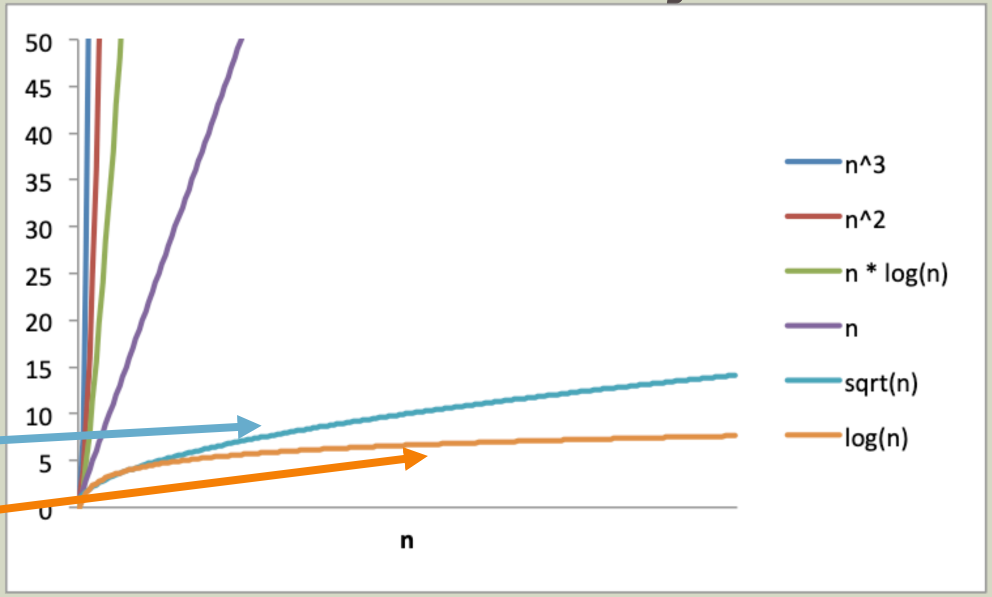
b) $f(n) = n^2$

c) $f(n) = n * \log(n)$

d) $f(n) = n$

e) $f(n) = \mathrm{sqrt}(n)$

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



COMPLEXIDADE DE ALGORITMOS

- Para **valores pequenos de *n***;
  - **qualquer algoritmo custa pouco.**
- **Análise de algoritmos** é **realizada**;
  - para **valores grandes de *n***.

COMPLEXIDADE DE ALGORITMOS

- Problema de complexidade 2^n;
  - computador que executa 10^9 (aproximadamente 1 bilhão) de operações por segundo:

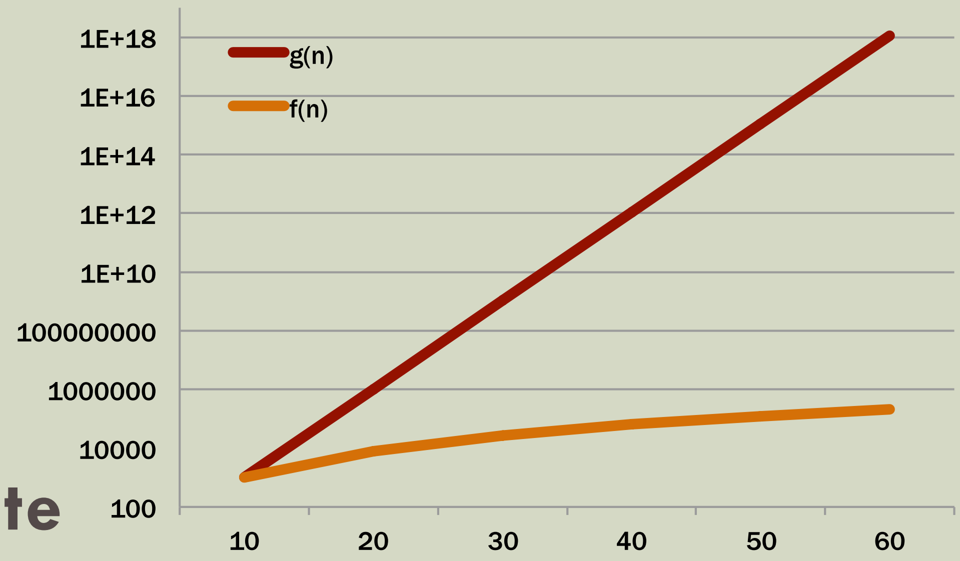
n	Tempo
10	0,0000001s
20	0,001s
30	1,07s
40	1099 s = 18 minutos e 20 s
50	18765 minutos = 13 dias
60	13344 dias ≅ 36,5 anos (!!!)

COMPORTAMENTO ASSINTÓTICO

- **Comportamento** das **funções de custo;**
  - para **valores grandes de *n*.**
- Representa o **limite do comportamento do custo;**
  - **quando *n* cresce.**

COMPORTAMENTO ASSINTÓTICO

- Dadas as funções:
  - $f(n) = n^3$
  - $g(n) = 2^n$
    - **$g(n)$ domina assintoticamente $f(n)$.**



NOTAÇÕES O, Ω e Θ

- Podem ser lidas como **aproximadamente.**
- Utilizaremos as **notações** O, Ω **e** Θ para **identificar** a **complexidade** (número aproximado de operações) **de um algoritmo.**

NOTAÇÕES O, Ω e Θ

- **Consideramos apenas a maior potência;**
  - ignoramos termos com menor crescimento.
- **Ignoramos** os **coeficientes;**
  - constantes.

NOTAÇÃO O

- **Limite assintótico superior do comportamento de um algoritmo.**
- Se uma **função** é $O(n^2)$;
  - ela **também** será **limitada assintoticamente por funções de graus superiores.**

NOTAÇÃO O

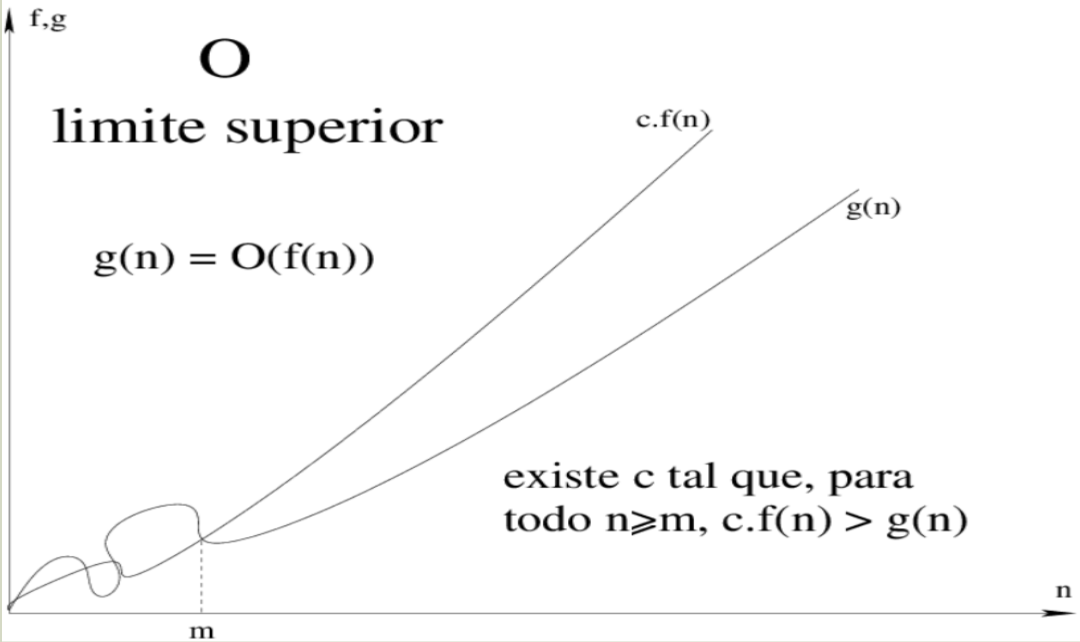
- Exemplo:
  - $g(n) = 3n^3 + 2n^2 + n$;
  - é $\mathbf{O(n^3)}$.

NOTAÇÃO O

- **$g(n) = O(f(n))$:**
  - $g(n)$ cresce, no máximo, tão rapidamente quanto $f(n)$;
  - $f(n)$ é um **limite** assintótico **superior** para $g(n)$;
  - $g(n)$ pode atingir $f(n)$;
    - mas nunca ultrapassá-la;
  - $f(n)$ limita $g(n)$ **por cima**;
  - $f(n)$ domina assintoticamente $g(n)$.

NOTAÇÃO O

- $g(n) = O(f(n))$;
  - se existirem constantes positivas c e *m* tais que;
    - para $n \geq m$, temos que $|g(n)| \leq c * |f(n)|$



EXERCÍCIO RESOLVIDO (2)

- Prove que $n^2 + 10 = O(n^2)$

EXERCÍCIO RESOLVIDO (2)

- Prove que $n^2 + 10 = O(n^2)$
  - $g(n) = n^2 + 10$
  - $f(n) = n^2$
  - $c = 2$
  - Agora precisamos descobrir um valor para m que valide a inequação: $0 \le n^2 + 10 \le 2n^2$.
    - $m = 1 \rightarrow 0 \le 11 \le 2$ (falso)
    - $m = 2 \rightarrow 0 \le 14 \le 8$ (falso)
    - $m = 3 \rightarrow 0 \le 19 \le 18$ (falso)
    - $m = 4 \rightarrow 0 \le 26 \le 32$ (verdadeiro)
  - Portanto, os valores $c = 2$ e $m = 4$ provam que:
    - $g(n) = n^2 + 10 = O(n^2)$.

EXERCÍCIO RESOLVIDO (3)

- Para cada função abaixo, identifique o limite superior O:
  - $f(n) = 3n^2 + 1$
  - $f(n) = 2n^3 + \lg n$
  - $f(n) = 5*n*\lg n + 2n$
  - $f(n) = 21$
  - $f(n) = \lg n + 3n$
  - $f(n) = \lg n + 2$

EXERCÍCIO RESOLVIDO (3)

- Para cada função abaixo, identifique o limite superior O:
  - $f(n) = 3n^2 + 1 = O(n^2), O(n^3), O(2^n)...$
  - $f(n) = 2n^3 + \lg n = O(n^3), O(n^4), O(2^n)...$
  - $f(n) = 5*n*\lg n + 2n = O(n*\lg n), O(n^2), O(n^3), O(2^n)...$
  - $f(n) = 21 = O(1), O(\lg n), O(n), O(n * \lg n)...$
    - As constantes não dependem do tamanho da entrada (n). Assim, convencionou-se que O(constante) = O(1)
  - $f(n) = \lg n + 3n = O(n), O(n^2), O(n^3), O(2^n)...$
  - $f(n) = \lg n + 2 = O(\lg n), O(n), O(n*\lg n), O(n^2), O(n^3), O(2^n)...$

NOTAÇÃO Ω

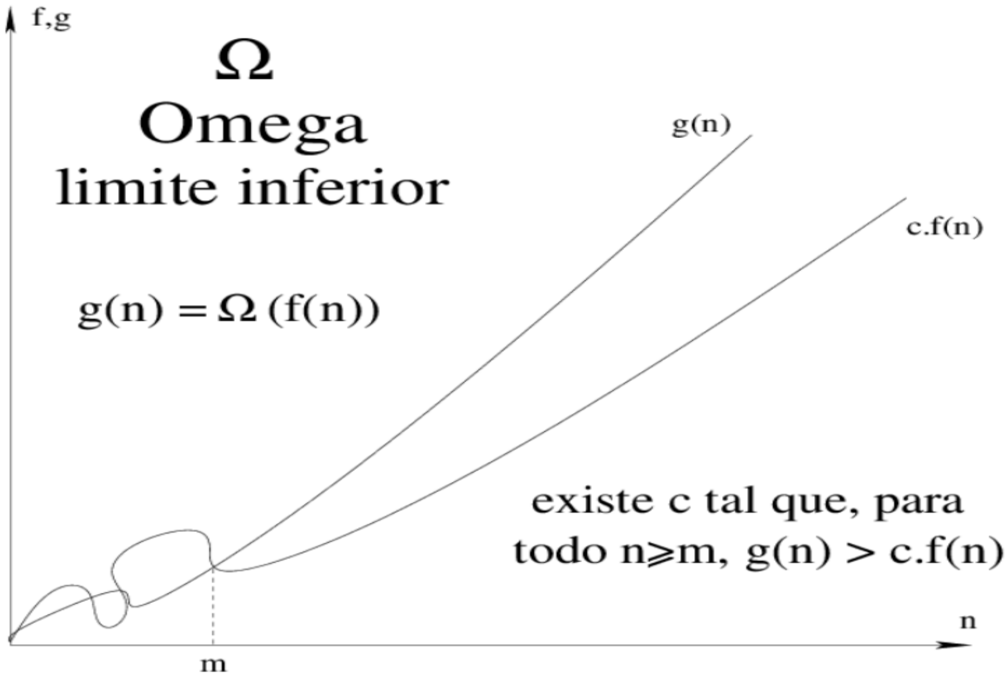
- **Limite assintótico inferior do comportamento de um algoritmo.**
- Se uma **função** é $\Omega(n^2)$;
  - ela **também** será **limitada assintoticamente por funções de graus inferiores.**

NOTAÇÃO Ω

- $\mathbf{g(n) = \Omega(f(n))}$:
  - $g(n)$ cresce, no mínimo, tão lentamente quanto $f(n)$;
  - $f(n)$ é um **limite** assintótico **inferior** para $g(n)$;
  - $f(n)$ limita $g(n)$ **por baixo.**

NOTAÇÃO Ω

- $g(n) = \Omega(f(n))$;
  - se existirem constantes positivas c e m tais que;
    - para $n \geq m$, temos que $|g(n)| \geq c * |f(n)|$



EXERCÍCIO RESOLVIDO (4)

- Prove que $n^2 + 10 = \Omega(n^2)$

EXERCÍCIO RESOLVIDO (4)

- Prove que $n^2 + 10 = \Omega(n^2)$
  - $g(n) = n^2 + 10$
  - $f(n) = n^2$
  - $c = 1$
  - Agora precisamos descobrir um valor para m que valide a inequação: $0 \leq n^2 \leq n^2 + 10$.
    - $m = 1 \rightarrow 0 \leq 1 \leq 11$ (verdadeiro)
    - $m = 2 \rightarrow 0 \leq 4 \leq 14$ (verdadeiro)
    - $m = 3 \rightarrow 0 \leq 9 \leq 19$ (verdadeiro)
    - $m = 4 \rightarrow 0 \leq 16 \leq 26$ (verdadeiro)
  - Portanto, com $c = 1$, qualquer valor de m que seja ≥ 0 prova que:
    - $g(n) = n^2 + 10 = \Omega(n^2)$.

EXERCÍCIO RESOLVIDO (5)

- Para cada função abaixo, identifique o limite inferior Ω:
  - $f(n) = 3n^2 + 1$
  - $f(n) = 2n^3 + \lg n$
  - $f(n) = 5*n*\lg n + 2n$
  - $f(n) = 21$
  - $f(n) = \lg n + 3n$
  - $f(n) = \lg n + 2$

EXERCÍCIO RESOLVIDO (5)

- Para cada função abaixo, identifique o limite inferior Ω:
  - $f(n) = 3n^2 + 1 = \Omega(n^2), \Omega(n*\lg n), \Omega(n), \Omega(\lg n), \Omega(1)$
  - $f(n) = 2n^3 + \lg n = \Omega(n^3), \Omega(n^2), \Omega(n), \Omega(\lg n), \Omega(1)$
  - $f(n) = 5*n*\lg n + 2n = \Omega(n*\lg n), \Omega(n), \Omega(\lg n), \Omega(1)$
  - $f(n) = 21 = \Omega(1)$
  - $f(n) = \lg n + 3n = \Omega(n), \Omega(\lg n), \Omega(1)$
  - $f(n) = \lg n + 2 = \Omega(\lg n), \Omega(1)$

NOTAÇÃO Θ

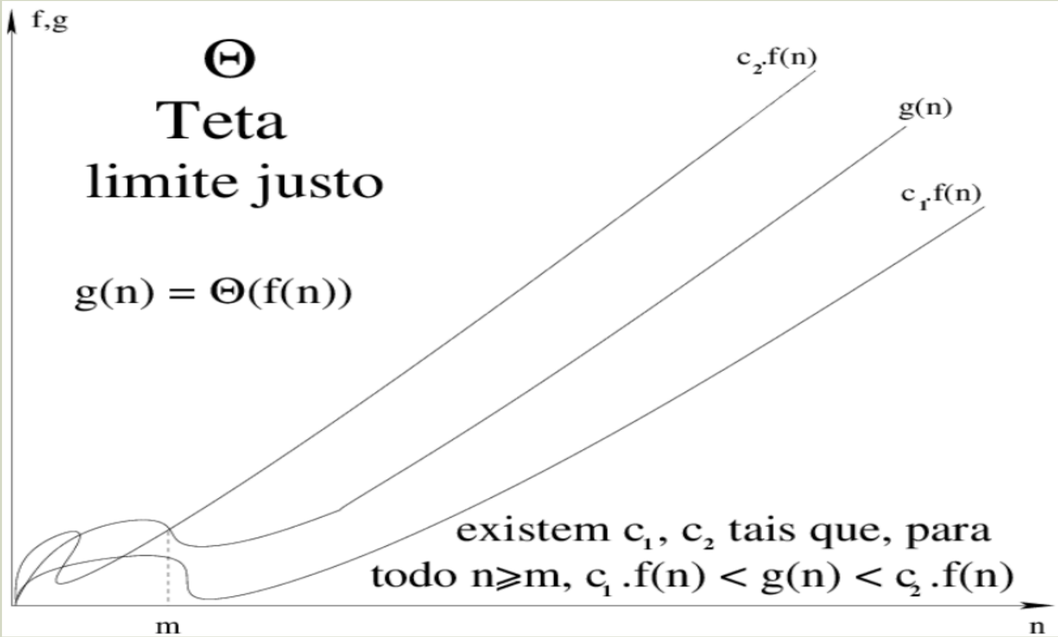
- **Limite assintótico justo.**
- $\boldsymbol{g(n) = \Theta(f(n))}$**:**
  - $g(n)$ cresce tão rapidamente quanto $f(n)$;
  - $f(n)$ **limita superiormente e inferiormente** $g(n)$;
  - $f(n)$ **limita** $g(n)$ **tanto por cima quanto por baixo.**
  - $f(n)$ é um limite assintótico restrito para $g(n)$.

NOTAÇÃO Θ

- Se $g(n)$ é $\Theta(f(n))$ então:
  - $g(n)$ é $O(f(n))$ e $\Omega(f(n))$.
- Se $g(n)$ é $O(f(n))$ e $\Omega(f(n))$ então:
  - $g(n)$ é $\Theta(f(n))$.

NOTAÇÃO Θ

- $g(n) = \Theta(f(n))$;
  - se existirem constantes positivas c_1, c_2 e m tais que;
    - para $n \geq m$, temos que $c_1 * |f(n)| \leq |g(n)| \leq c_2 * |f(n)|$



EXERCÍCIO RESOLVIDO (6)

- Prove que $n^2 + 10 = \Theta(n^2)$

EXERCÍCIO RESOLVIDO (6)

- Prove que $n^2 + 10 = \Theta(n^2)$
  - $g(n) = n^2 + 10$
  - $f(n) = n^2$
  - $c_1 = 1$ e $c_2 = 2$
  - Agora precisamos descobrir um valor para m que valide a inequação: $0 \leq n^2 \leq n^2 + 10 \leq 2n^2$.
    - $m = 1 \rightarrow 0 \leq 1 \leq 11 \leq 2$ (falso)
    - $m = 2 \rightarrow 0 \leq 4 \leq 14 \leq 8$ (falso)
    - $m = 3 \rightarrow 0 \leq 9 \leq 19 \leq 18$ (falso)
    - $m = 4 \rightarrow 0 \leq 16 \leq 26 \leq 32$ (verdadeiro)
  - Portanto, os valores $c_1 = 1$, $c_2 = 2$ e $m = 4$ provam que:
    - $g(n) = n^2 + 10 = \Theta(n^2)$.

EXERCÍCIO RESOLVIDO (7)

- Para cada função abaixo, identifique o Θ:
  - $f(n) = 3n^2 + 1$
  - $f(n) = 2n^3 + \lg n$
  - $f(n) = 5*n*\lg n + 2n$
  - $f(n) = 21$
  - $f(n) = \lg n + 3n$
  - $f(n) = \lg n + 2$

EXERCÍCIO RESOLVIDO (7)

- Para cada função abaixo, identifique o Θ:
  - $f(n) = 3n^2 + 1 = \Theta(n^2)$
  - $f(n) = 2n^3 + \lg n = \Theta(n^3)$
  - $f(n) = 5*n*\lg n + 2n = \Theta(n*\lg n)$
  - $f(n) = 21 = \Theta(1)$
  - $f(n) = \lg n + 3n = \Theta(n)$
  - $f(n) = \lg n + 2 = \Theta(\lg n)$

EXERCÍCIO RESOLVIDO (8)

- Responda se as afirmações abaixo são verdadeiras ou falsas:
  - $3n^2 + 5n + 1$ é $O(n)$
  - $3n^2 + 5n + 1$ é $O(n^2)$
  - $3n^2 + 5n + 1$ é $O(n^3)$
  - $3n^2 + 5n + 1$ é $\Omega(n)$
  - $3n^2 + 5n + 1$ é $\Omega(n^2)$
  - $3n^2 + 5n + 1$ é $\Omega(n^3)$
  - $3n^2 + 5n + 1$ é $\Theta(n)$
  - $3n^2 + 5n + 1$ é $\Theta(n^2)$
  - $3n^2 + 5n + 1$ é $\Theta(n^3)$

EXERCÍCIO RESOLVIDO (8)

- Responda se as afirmações abaixo são verdadeiras ou falsas:
  - $3n^2 + 5n + 1$ é $O(n)$ (falso)
  - $3n^2 + 5n + 1$ é $O(n^2)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $O(n^3)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $\Omega(n)$
  - $3n^2 + 5n + 1$ é $\Omega(n^2)$
  - $3n^2 + 5n + 1$ é $\Omega(n^3)$
  - $3n^2 + 5n + 1$ é $\Theta(n)$
  - $3n^2 + 5n + 1$ é $\Theta(n^2)$
  - $3n^2 + 5n + 1$ é $\Theta(n^3)$

EXERCÍCIO RESOLVIDO (8)

- Responda se as afirmações abaixo são verdadeiras ou falsas:
  - $3n^2 + 5n + 1$ é $O(n)$ (falso)
  - $3n^2 + 5n + 1$ é $O(n^2)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $O(n^3)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $\Omega(n)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $\Omega(n^2)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $\Omega(n^3)$ (falso)
  - $3n^2 + 5n + 1$ é $\Theta(n)$
  - $3n^2 + 5n + 1$ é $\Theta(n^2)$
  - $3n^2 + 5n + 1$ é $\Theta(n^3)$

EXERCÍCIO RESOLVIDO (8)

- Responda se as afirmações abaixo são verdadeiras ou falsas:
  - $3n^2 + 5n + 1$ é $O(n)$ (falso)
  - $3n^2 + 5n + 1$ é $O(n^2)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $O(n^3)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $\Omega(n)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $\Omega(n^2)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $\Omega(n^3)$ (falso)
  - $3n^2 + 5n + 1$ é $\Theta(n)$ (falso)
  - $3n^2 + 5n + 1$ é $\Theta(n^2)$ (verdadeiro)
  - $3n^2 + 5n + 1$ é $\Theta(n^3)$ (falso)

EXERCÍCIO RESOLVIDO (9)

- Preencha a tabela abaixo com verdadeiro ou falso:

	O(1)	O(lg *n*)	O(*n*)	O(*n**lg(*n*))	O(n^2)	O(n^3)	O(n^5)	O(n^{20})
$g(n) = \lg(n)$								
$g(n) = n * \lg(n)$								
$g(n) = 5n + 1$								
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (9)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$O(1)$	$O(\lg n)$	$O(n)$	$O(n*\lg(n))$	$O(n^2)$	$O(n^3)$	$O(n^5)$	$O(n^{20})$
$g(n) = \lg(n)$	F	V	V	V	V	V	V	V
$g(n) = n * \lg(n)$								
$g(n) = 5n + 1$								
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (9)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$O(1)$	$O(\lg n)$	$O(n)$	$O(n*\lg(n))$	$O(n^2)$	$O(n^3)$	$O(n^5)$	$O(n^{20})$
$g(n) = \lg(n)$	F	V	V	V	V	V	V	V
$g(n) = n * \lg(n)$	F	F	F	V	V	V	V	V
$g(n) = 5n + 1$								
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (9)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$O(1)$	$O(\lg n)$	$O(n)$	$O(n*\lg(n))$	$O(n^2)$	$O(n^3)$	$O(n^5)$	$O(n^{20})$
$g(n) = \lg(n)$	F	V	V	V	V	V	V	V
$g(n) = n * \lg(n)$	F	F	F	V	V	V	V	V
$g(n) = 5n + 1$	F	F	V	V	V	V	V	V
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (9)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$O(1)$	$O(\lg n)$	$O(n)$	$O(n*\lg(n))$	$O(n^2)$	$O(n^3)$	$O(n^5)$	$O(n^{20})$
$g(n) = \lg(n)$	F	V	V	V	V	V	V	V
$g(n) = n * \lg(n)$	F	F	F	V	V	V	V	V
$g(n) = 5n + 1$	F	F	V	V	V	V	V	V
$g(n) = 7n^5 - 3n^2$	F	F	F	F	F	F	V	V
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (9)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$O(1)$	$O(\lg n)$	$O(n)$	$O(n*\lg(n))$	$O(n^2)$	$O(n^3)$	$O(n^5)$	$O(n^{20})$
$g(n) = \lg(n)$	F	V	V	V	V	V	V	V
$g(n) = n * \lg(n)$	F	F	F	V	V	V	V	V
$g(n) = 5n + 1$	F	F	V	V	V	V	V	V
$g(n) = 7n^5 - 3n^2$	F	F	F	F	F	F	V	V
$g(n) = 99n^3 - 1000n^2$	F	F	F	F	F	V	V	V
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (9)

- Preencha a tabela abaixo com verdadeiro ou falso:

	O(1)	O(lg n)	O(n)	O(n*lg(n))	O(n^2)	O(n^3)	O(n^5)	O(n^{20})
$g(n) = \lg(n)$	F	V	V	V	V	V	V	V
$g(n) = n * \lg(n)$	F	F	F	V	V	V	V	V
$g(n) = 5n + 1$	F	F	V	V	V	V	V	V
$g(n) = 7n^5 - 3n^2$	F	F	F	F	F	F	V	V
$g(n) = 99n^3 - 1000n^2$	F	F	F	F	F	V	V	V
$g(n) = n^5 - 99999n^4$	F	F	F	F	F	F	V	V

EXERCÍCIO RESOLVIDO (10)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Omega(1)$	$\Omega(\lg n)$	$\Omega(n)$	$\Omega(n*\lg(n))$	$\Omega(n^2)$	$\Omega(n^3)$	$\Omega(n^5)$	$\Omega(n^{20})$
$g(n) = \lg(n)$								
$g(n) = n * \lg(n)$								
$g(n) = 5n + 1$								
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (10)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Omega(1)$	$\Omega(\lg n)$	$\Omega(n)$	$\Omega(n*\lg(n))$	$\Omega(n^2)$	$\Omega(n^3)$	$\Omega(n^5)$	$\Omega(n^{20})$
$g(n) = \lg(n)$	**V**	**V**	**F**	**F**	**F**	**F**	**F**	**F**
$g(n) = n * \lg(n)$								
$g(n) = 5n + 1$								
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (10)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Omega(1)$	$\Omega(\lg n)$	$\Omega(n)$	$\Omega(n*\lg(n))$	$\Omega(n^2)$	$\Omega(n^3)$	$\Omega(n^5)$	$\Omega(n^{20})$
$g(n) = \lg(n)$	V	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	V	V	V	V	F	F	F	F
$g(n) = 5n + 1$								
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (10)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Omega(1)$	$\Omega(\lg n)$	$\Omega(n)$	$\Omega(n*\lg(n))$	$\Omega(n^2)$	$\Omega(n^3)$	$\Omega(n^5)$	$\Omega(n^{20})$
$g(n) = \lg(n)$	V	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	V	V	V	V	F	F	F	F
$g(n) = 5n + 1$	V	V	V	F	F	F	F	F
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (10)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Omega(1)$	$\Omega(\lg n)$	$\Omega(n)$	$\Omega(n*\lg(n))$	$\Omega(n^2)$	$\Omega(n^3)$	$\Omega(n^5)$	$\Omega(n^{20})$
$g(n) = \lg(n)$	V	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	V	V	V	V	F	F	F	F
$g(n) = 5n + 1$	V	V	V	F	F	F	F	F
$g(n) = 7n^5 - 3n^2$	V	V	V	V	V	V	V	F
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (10)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Omega(1)$	$\Omega(\lg n)$	$\Omega(n)$	$\Omega(n*\lg(n))$	$\Omega(n^2)$	$\Omega(n^3)$	$\Omega(n^5)$	$\Omega(n^{20})$
$g(n) = \lg(n)$	V	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	V	V	V	V	F	F	F	F
$g(n) = 5n + 1$	V	V	V	F	F	F	F	F
$g(n) = 7n^5 - 3n^2$	V	V	V	V	V	V	V	F
$g(n) = 99n^3 - 1000n^2$	V	V	V	V	V	V	F	F
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (10)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Omega(1)$	$\Omega(\lg n)$	$\Omega(n)$	$\Omega(n*\lg(n))$	$\Omega(n^2)$	$\Omega(n^3)$	$\Omega(n^5)$	$\Omega(n^{20})$
$g(n) = \lg(n)$	V	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	V	V	V	V	F	F	F	F
$g(n) = 5n + 1$	V	V	V	F	F	F	F	F
$g(n) = 7n^5 - 3n^2$	V	V	V	V	V	V	V	F
$g(n) = 99n^3 - 1000n^2$	V	V	V	V	V	V	F	F
$g(n) = n^5 - 99999n^4$	V	V	V	V	V	V	V	F

EXERCÍCIO RESOLVIDO (11)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Theta(1)$	$\Theta(\lg n)$	$\Theta(n)$	$\Theta(n*\lg(n))$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^5)$	$\Theta(n^{20})$
$g(n) = \lg(n)$								
$g(n) = n * \lg(n)$								
$g(n) = 5n + 1$								
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (11)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Theta(1)$	$\Theta(\lg n)$	$\Theta(n)$	$\Theta(n*\lg(n))$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^5)$	$\Theta(n^{20})$
$g(n) = \lg(n)$	F	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$								
$g(n) = 5n + 1$								
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (11)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Theta(1)$	$\Theta(\lg n)$	$\Theta(n)$	$\Theta(n*\lg(n))$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^5)$	$\Theta(n^{20})$
$g(n) = \lg(n)$	F	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	F	F	F	V	F	F	F	F
$g(n) = 5n + 1$								
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (11)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Theta(1)$	$\Theta(\lg n)$	$\Theta(n)$	$\Theta(n*\lg(n))$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^5)$	$\Theta(n^{20})$
$g(n) = \lg(n)$	F	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	F	F	F	V	F	F	F	F
$g(n) = 5n + 1$	F	F	V	F	F	F	F	F
$g(n) = 7n^5 - 3n^2$								
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (11)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Theta(1)$	$\Theta(\lg n)$	$\Theta(n)$	$\Theta(n*\lg(n))$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^5)$	$\Theta(n^{20})$
$g(n) = \lg(n)$	F	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	F	F	F	V	F	F	F	F
$g(n) = 5n + 1$	F	F	V	F	F	F	F	F
$g(n) = 7n^5 - 3n^2$	F	F	F	F	F	F	V	F
$g(n) = 99n^3 - 1000n^2$								
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (11)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Theta(1)$	$\Theta(\lg n)$	$\Theta(n)$	$\Theta(n*\lg(n))$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^5)$	$\Theta(n^{20})$
$g(n) = \lg(n)$	F	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	F	F	F	V	F	F	F	F
$g(n) = 5n + 1$	F	F	V	F	F	F	F	F
$g(n) = 7n^5 - 3n^2$	F	F	F	F	F	F	V	F
$g(n) = 99n^3 - 1000n^2$	F	F	F	F	F	V	F	F
$g(n) = n^5 - 99999n^4$								

EXERCÍCIO RESOLVIDO (11)

- Preencha a tabela abaixo com verdadeiro ou falso:

	$\Theta(1)$	$\Theta(\lg n)$	$\Theta(n)$	$\Theta(n*\lg(n))$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^5)$	$\Theta(n^{20})$
$g(n) = \lg(n)$	F	V	F	F	F	F	F	F
$g(n) = n * \lg(n)$	F	F	F	V	F	F	F	F
$g(n) = 5n + 1$	F	F	V	F	F	F	F	F
$g(n) = 7n^5 - 3n^2$	F	F	F	F	F	F	V	F
$g(n) = 99n^3 - 1000n^2$	F	F	F	F	F	V	F	F
$g(n) = n^5 - 99999n^4$	F	F	F	F	F	F	V	F

EXERCÍCIO RESOLVIDO (12)

- Indique o tipo de crescimento que melhor caracteriza as funções abaixo (*Khan Academy*, adaptado):

	Constante	**Linear**	**Polinomial**	**Exponencial**
$3n$				
1				
$(3/2)n$				
$2n^3$				
2^n				
$3n^2$				
1000				
$(3/2)^n$				

EXERCÍCIO RESOLVIDO (12)

- Indique o tipo de crescimento que melhor caracteriza as funções abaixo (*Khan Academy*, adaptado):

	Constante	Linear	Polinomial	Exponencial
$3n$		✓		
1	✓			
$(3/2)n$		✓		
$2n^3$			✓	
2^n				✓
$3n^2$			✓	
1000	✓			
$(3/2)^n$				✓

EXERCÍCIO RESOLVIDO (13)

- Classifique as funções $f_1(n) = n^2$, $f_2(n) = n$, $f_3(n) = 2^n$, $f_4(n) = (3/2)^n$, $f_5(n) = n^3$ e $f_6(n) = 1$ de acordo com seu crescimento, do mais lento para o mais rápido (*Khan Academy*, adaptado).

EXERCÍCIO RESOLVIDO (13)

- Classifique as funções $f_1(n) = n^2$, $f_2(n) = n$, $f_3(n) = 2^n$, $f_4(n) = (3/2)^n$, $f_5(n) = n^3$ e $f_6(n) = 1$ de acordo com seu crescimento, do mais lento para o mais rápido (*Khan Academy*, adaptado).
  - $f_6(n) = 1$
  - $f_2(n) = n$
  - $f_1(n) = n^2$
  - $f_5(n) = n^3$
  - $f_4(n) = (3/2)^n$
  - $f_3(n) = 2^n$

EXERCÍCIO RESOLVIDO (14)

- Classifique as funções $f_1(n) = n*\log_6 n$, $f_2(n) = \lg n$, $f_3(n) = \log_8 n$, $f_4(n) = 8n^2$, $f_5(n) = n*\lg n$, $f_6(n) = 64$, $f_7(n) = 6n^3$, $f_8(n) = 8^{2n}$ e $f_9(n) = 4n$ de acordo com seu crescimento, do mais lento para o mais rápido (*Khan Academy*, adaptado).

EXERCÍCIO RESOLVIDO (14)

- Classifique as funções $f_1(n) = n*\log_6 n$, $f_2(n) = \lg n$, $f_3(n) = \log_8 n$, $f_4(n) = 8n^2$, $f_5(n) = n*\lg n$, $f_6(n) = 64$, $f_7(n) = 6n^3$, $f_8(n) = 8^{2n}$ e $f_9(n) = 4n$ de acordo com seu crescimento, do mais lento para o mais rápido (*Khan Academy*, adaptado).
  - $f_6(n) = 64$
  - $f_3(n) = \log_8 n$
  - $f_2(n) = \lg n$
  - $f_9(n) = 4n$
  - $f_1(n) = n*\log_6 n$
  - $f_5(n) = n*\lg n$
  - $f_4(n) = 8n^2$
  - $f_7(n) = 6n^3$
  - $f_8(n) = 8^{2n}$

✓

EXERCÍCIO RESOLVIDO (15)

- Faça a correspondência entre cada função $g(n)$ com sua $f(n)$ equivalente, em termos de Θ. Essa correspondência ocorre quando $g(n) = \Theta(f(n))$ (*Khan Academy*, adaptado).

$g(n)$	**$f(n)$**
$n + 30$	n^4
$n^2 + 2n - 10$	$3n - 1$
$n^3 * 3n$	$\lg(2n)$
$\lg(n)$	$n^2 + 3n$

EXERCÍCIO RESOLVIDO (15)

- Faça a correspondência entre cada função $g(n)$ com sua $f(n)$ equivalente, em termos de Θ. Essa correspondência ocorre quando $g(n) = \Theta(f(n))$ (*Khan Academy*, adaptado).

$g(n)$	**$f(n)$**
$n + 30$	n^4
$n^2 + 2n - 10$	$3n - 1$
$n^3 * 3n$	$\lg(2n)$
$\lg(n)$	$n^2 + 3n$

EXERCÍCIO RESOLVIDO (16)

- Apresente a função e a complexidade, em termos de Θ, para o número de subtrações do pior e melhor casos.

```
i = 0;
while (i < n) {
        i++;
        a--; }
if (b > c) {
        i--;
} else {
        i--;
        a--; }
```

EXERCÍCIO RESOLVIDO (16)

- Apresente a função e a complexidade, em termos de Θ, para o número de subtrações do pior e melhor casos.

```
i = 0;
while (i < n) {
        i++;
        a--; }
if (b > c) {
        i--;
} else {
        i--;
        a--; }
```

- Pior caso:
  - função de complexidade: $n + 2$
  - $\Theta(n)$
- Melhor caso:
  - função de complexidade: $n + 1$
  - $\Theta(n)$

EXERCÍCIO RESOLVIDO (17)

- Apresente a função e a complexidade, em termos de Θ, para o número de subtrações do pior e melhor casos.

```
for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        a--;
        b--;
    }
    c--;
}
```

EXERCÍCIO RESOLVIDO (17)

- Apresente a função e a complexidade, em termos de Θ, para o número de subtrações do pior e melhor casos.

```
for (i = 0; i < n; i++) {
      for (j = 0; j < n; j++) {
            a--;
            b--;
      }
      c--;
}
```

- Pior e melhor casos:
  - função de complexidade: $(2n + 1)*n$
  - $\Theta(n^2)$

EXERCÍCIO RESOLVIDO (18)

- Apresente a função e a complexidade, em termos de Θ, para o número de subtrações do pior e melhor casos.

```
for (i = 0; i < n; i++) {
    for (j = 1; j <= n; j*=2) {
        b--;
    }
}
```

EXERCÍCIO RESOLVIDO (18)

- Apresente a função e a complexidade, em termos de Θ, para o número de subtrações do pior e melhor casos.

```
for (i = 0; i < n; i++) {
      for (j = 1; j <= n; j*=2) {
            b--;
      }
}
```

- Pior e melhor casos:
  - função de complexidade: $(\lg(n) + 1)*n = n*\lg(n) + n$
  - $\Theta(n*\lg(n))$

EXERCÍCIO RESOLVIDO (19)

- Apresente a função e a complexidade, em termos de Θ, para o número de comparações e movimentações do pior e melhor casos.

EXERCÍCIO RESOLVIDO (19)

```
void imprimirMaxMin( int [] array, int n){
        int maximo, minimo;
        if (array[0] > array[1])
            maximo = array[0];  minimo = array[1];
        else
            maximo = array[1];  minimo = array[0];
        for (int i = 2; i < n; i++)
            if (array[i] > maximo)
                maximo = array[i];
            else if (array[i] < minimo)
                minimo = array[i];
}
```

EXERCÍCIO RESOLVIDO (19)

- Número de comparações:
  - Pior caso:
    - função de complexidade: $1 + 2*(n - 2)$
    - $\Theta(n)$
  - Melhor caso:
    - função de complexidade: $1 + 1*(n - 2)$
    - $\Theta(n)$

EXERCÍCIO RESOLVIDO (19)

- Número de movimentações:
  - Pior caso:
    - função de complexidade: $2 + 1*(n - 2)$
    - $\Theta(n)$
  - Melhor caso:
    - função de complexidade: $2 + 0*(n - 2)$
    - $\Theta(1)$

CLASSES DE COMPLEXIDADE

- **Complexidade constante** ou fixa:
  - $f(n) = O(1)$.
  - **Tempo de execução do algoritmo;**
    - **independe do tamanho da entrada:** n.
  - **Instruções do algoritmo;**
    - **executadas** um **número fixo de vezes.**

CLASSES DE COMPLEXIDADE

- **Complexidade logarítmica:**
  - $f(n) = O(\lg n)$.
  - Tipicamente, é a **complexidade de algoritmos** que **resolvem problemas;**
    - **transformando-os** em **problemas menores.**
  - Exemplo:
    - algoritmo de Pesquisa Binária.

CLASSES DE COMPLEXIDADE

- **Complexidade logarítmica:**
  - $f(n) = O(\lg n)$.
  - Supondo que a base do logaritmo seja 2:
    - para $n = 1.000$;
      - $f(1.000) = \lg 1.000 \cong 10$;
    - para $n = 1.000.000$;
      - $f(1.000.000) = \lg 1.000.000 \cong 20$.

CLASSES DE COMPLEXIDADE

- **Complexidade linear:**
  - $f(n) = O(n)$.
  - Um **pequeno trabalho** é **realizado**;
    - **sobre cada elemento da entrada.**
  - Cada vez que ***n* dobra de tamanho**;
    - o **tempo de execução** do algoritmo também **dobra**.
  - Exemplo:
    - algoritmo de Pesquisa Sequencial.

CLASSES DE COMPLEXIDADE

- **Algoritmos *n* * lg *n*:**
  - $f(n) = O(n \lg n)$.
  - Tipicamente, é a **complexidade de algoritmos** que **resolvem problemas;**
    - **quebrando-os** em **problemas menores;**
    - **resolvendo cada um deles independentemente;**
    - **agrupando** as **soluções.**

CLASSES DE COMPLEXIDADE

- **Algoritmos *n* * lg *n*:**
  - **$f(n) = O(n \lg n)$.**
  - Supondo que a base do logaritmo seja 2:
    - para $n = 1.000.000$;
      - $f(1.000.000) = 1.000.000 * \lg 1.000.000 \cong 20.000.000$;
    - para $n = 2.000.000$;
      - $f(2.000.000) = 2.000.000 * \lg 2.000.000 \cong 42.000.000$.

CLASSES DE COMPLEXIDADE

- **Complexidade quadrática:**
  - $f(n) = O(n^2)$.
  - **Itens de dados** são **processados aos pares;**
    - tipicamente, em **um laço dentro de outro.**
  - Algoritmos desta classe são úteis para resolver problemas;
    - de tamanho relativamente pequeno.

CLASSES DE COMPLEXIDADE

- **Complexidade quadrática:**
  - $f(n) = O(n^2)$.

n	n^2
10	100
20	400
30	900
40	1600
50	2500
60	3600

CLASSES DE COMPLEXIDADE

- **Complexidade cúbica:**
  - $f(n) = O(n^3)$.
  - Cada vez que ***n*** **dobra de tamanho**;
    - o **tempo de execução** do algoritmo é **multiplicado por 8.**
  - Algoritmos desta classe são úteis para resolver problemas;
    - de tamanho relativamente pequeno.
  - Exemplo:
    - algoritmo para Multiplicação de Matrizes.

CLASSES DE COMPLEXIDADE

- **Complexidade exponencial:**
  - $f(n) = O(2^n)$.
  - Cada vez que ***n* dobra de tamanho**;
    - o **tempo de execução** do algoritmo é **elevado ao quadrado.**
  - Algoritmos desta classe;
    - não são úteis sob o ponto de vista prático.
  - Ocorre na **solução de problemas por força bruta.**

CLASSES DE COMPLEXIDADE

- **Complexidade fatorial:**
  - $f(n) = O(n!)$.
  - Também ocorre na **solução de problemas por força bruta;**
    - com a identificação de **todas as combinações possíveis para a resolução do problema.**
  - Comportamento muito pior do que $O(2^n)$.

CLASSES DE COMPLEXIDADE

- **Complexidade fatorial:**
  - $\boldsymbol{f(n) = O(n!)}$.
  - Para n = 20;
    - $f(20) = 20! = 2432902008176640000$;
      - um número de 19 dígitos!
  - Para n = 40;
    - $f(40) = 40! \cong 8{,}16 \times 10^{47}$;
      - um número de 48 dígitos!

CLASSES DE COMPLEXIDADE

- **Complexidade fatorial:**
  - $f(n) = O(n!)$.
  - $f(40) = 40! \cong 8{,}16 \times 10^{47}$.
    - Um computador que execute 1 bilhão de operações por segundo;
      - levaria 10^{38} segundos para resolver este problema.
        - 10^{38} segundos = 31.709.791.983.764.586.504.312.531.709,792 séculos!

TEMPO DE EXECUÇÃO ESTIMADO

Função de custo	Tamanho da entrada n					
	10	20	30	40	50	60
n	0,00001s	0,00002s	0,00003s	0,00004s	0,00005s	0,00006s
n^2	0,0001s	0,0004s	0,0009s	0,0016s	0,0025s	0,0036s
n^3	0,001s	0,008s	0,027s	0,064s	0,125s	0,316s
n^5	0,1s	3,2s	24,3s	1,7 minutos	5,2 minutos	13 minutos
2^n	0,001s	1s	17,9 minutos	12,7 dias	35,7 anos	366 séculos
3^n	0,059s	58 minutos	6,5 anos	3855 séculos	10^8 séculos	10^{13} séculos

ALGORITMOS POLINOMIAIS X ALGORITMOS EXPONENCIAIS

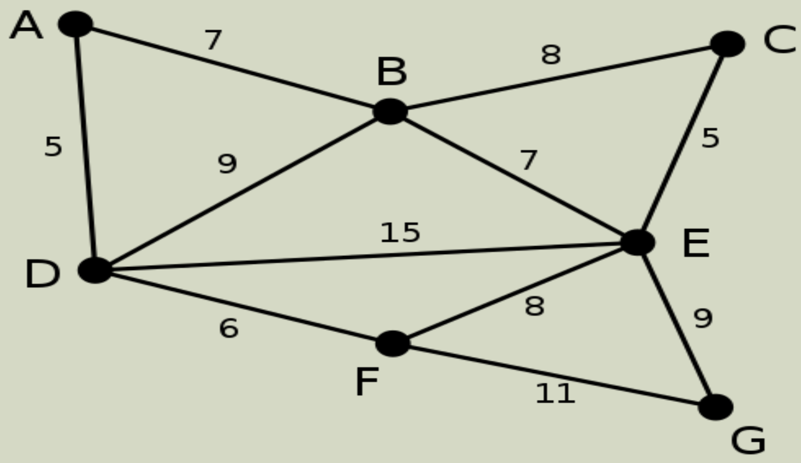
- **Algoritmo polinomial**:
  - algoritmo cuja **função de complexidade** é **O(n^p)**;
    - sendo ***p*** um inteiro correspondente ao **grau do polinômio.**
- **Algoritmo exponencial**:
  - algoritmo cuja **função de complexidade** é **O(c^n)**;
    - sendo **$c > 1$**.

ALGORITMOS POLINOMIAIS X ALGORITMOS EXPONENCIAIS

- Um **problema** é **considerado intratável;**
  - quando ele é tão difícil que **não existe algoritmo polinomial capaz de resolvê-lo.**
- Um **problema** é considerado **bem resolvido;**
  - quando **existe** algum **algoritmo polinomial capaz de resolvê-lo.**

ALGORITMOS EXPONENCIAIS

- Um algoritmo que leva **séculos para finalizar sua execução;**
  - **não** é uma **opção adequada.**
- **Problema do Caixeiro Viajante:**



PROBLEMA DO CAIXEIRO VIAJANTE

Número de Cidades	Tempo de Execução
5	5 s
6	5 * 5 = 25 s
7	6 * 25 = 150 s = 2,5 minutos
8	7 * 2,5 = 17,5 minutos
9	8 * 17,5 = 140 minutos = 2,34 horas
10	9 * 2,34 = 21 horas
11	10 * 21 = 210 horas = 8,75 dias
12	11 * 8,75 = 96,25 dias
13	12 * 96,25 = 1155 dias = 3,15 anos
14	13 * 3,15 = 41,02 anos
15	14 * 41,02 = 574 anos
16	15 * 574 = 8,6 séculos

ALGORITMOS POLINOMIAIS X ALGORITMOS EXPONENCIAIS

- **Algoritmos eficientes:**
  - **tempo polinomial.**
- Algoritmos **ineficientes:**
  - **tempo superpolinomial.**