Pesquisa e Classificação de Dados
Lista 1 (Análise de Algoritmos)

Prof. Ricardo Oliveira

OBS: Apenas o resultado final é apresentado. Realize também o desenvolvimento do exercício!

OBS2: Caso encontre algum erro, por favor avise o professor.

1. Note que outros valores para c e n_0 também são válidos!
   (a) $c = 1, n_0 = 0$
   (b) $c = 4, n_0 = 1$
   (c) $c = 3, n_0 = 1$
   (d) $c = 1, n_0 = 0$
   (e) $c = 324, n_0 = 42$
   (f) $c = 1, n_0 = 2$
   (g) $c = 2, n_0 = 2$
2. Por contradição: considere a hipótese que $n^2 = O(n)$. Logo, existem $c > 0, n_0 \in \mathbb{N}$ tais que $n^2 \leq cn$ para todo $n \geq n_0$. Seja n_x qualquer inteiro maior que c e n_0 simultaneamente, como, por exemplo, $n_x = max\{\lceil c \rceil, n_0\}+1$. Como:
   $n^2 \leq cn$ para todo $n \geq n_0$ (hipótese); e
   $n_x \geq n_0$ (definição de n_x)
   tem-se que:
   $n_x^2 \leq cn_x$
   Desta equação, temos que $n_x \leq c$. Isto é uma contradição, pois $n_x > c$ por definição. ■
3. Como $f(n) = O(g(n))$, existem $c_1 > 0, n_1 \in \mathbb{N}$ tais que $f(n) \leq c_1 g(n)$ para todo $n \geq n_1$. Como $g(n) = O(p(n))$, existem $c_2 > 0, n_2 \in \mathbb{N}$ tais que $g(n) \leq c_2 p(n)$ para todo $n \geq n_2$. Desta forma, para todo $n \geq max\{n_1, n_2\}$ tem-se $f(n) \leq c_1 g(n) \leq c_1(c_2 p(n))$. Logo, tome $c = c_1 c_2$ e $n_0 = max\{n_1, n_2\}$. ■
4. Como $f(n) = O(\log_2 n)$, existem $c_1 > 0, n_0 \in \mathbb{N}$ tais que $f(n) \leq c_1 \log_2 n$ para todo $n \geq n_0$. Seja $b > 2$ uma base. Como $\log_2 n = \log_b n / \log_b 2$, tem-se $f(n) \leq c_1(\log_b n / \log_b 2)$. Logo, tome $c = c_1 / \log_b 2$ e mantenha n_0. ■
5. (a) $O(1)$
   (b) $O(n)$
   (c) $O(n^3)$
6. (a) $c = 1, n_0 = 0$
   (b) $c = 1, n_0 = 20$
   (c) $c = 1, n_0 = 1$
7. $2^{n+1} = 2^n \times 2 = \Omega(2^n)(c = 2, n_0 = 0)$. Além disso, $3^n = \Omega(2^n)$ pois $2^n = O(3^n)$ $(c = 1, n_0 = 1)$.
8. (a) $3n + 4 = O(n)$ com $c = 4, n_0 = 4$; $n = O(3n + 4)$ com $c = 1, n_0 = 1$.

(b) $n \log n + n^2 + 3 = O(n^2)$ pelos teorema 2. $n^2 = O(n \log n + n^2 + 3)$ com $c = 1, n_0 = 1$.
(c) $2^n + n^2 = O(2^n)$ com $c = 2, n_0 = 4$. $2^n = O(2^n + n^2)$ com $c = 1, n_0 = 0$.

9. (a) Pior caso = Melhor caso = $O(n)$
(b) Pior caso = Melhor caso = $O(n)$
(c) Pior caso =$O(\sqrt{n})$, Melhor caso = $O(1)$
(d) Invariante: após i laços, $pot = 2^i$. O algoritmo para com $pot \geq n = 2^i \geq n = i \geq \log_2 n$. Logo, Pior caso = Melhor caso = $O(\log_2 n)$.
(e) Pior caso = Melhor caso = $O(n) + O(m) = O(n + m)$ [1].
(f) Pior caso = Melhor caso = $O(\log_2 \exp)$
(g) Pior caso = Melhor caso = $O(3^n)$

10. (a) Dica: Use o método da árvore de recorrência e o fato de que $T(n-2)$ é sempre menor ou igual a $T(n-1)$.
(b) Dica: Expanda a recorrência e conclua o "padrão" de que o número de operações é proporcinal a $fib(n)$, o próprio $n-$ésimo número de Fibonacci. Por fim, utilize a fórmula de Binet[2] para concluir que $fib(n) = \Theta(\phi^n)$.

11. Melhor caso = $O(1)$; Pior caso = Caso Médio = $O(N)$
12. Pior caso = Melhor Caso = Caso Médio = $\Theta(N)$
13. $O(1)$

[1] *https://cs.stackexchange.com/questions/30036/can-a-big-oh-time-complexity-contain-more-than-one-variable*
[2] *http://mathworld.wolfram.com/BinetsFibonacciNumberFormula.html*