

ANÁLISE DE ALGORITMOS

Questão 1.

1. $O(n^n)$ (Exponencial, muito rápido crescimento)
2. $O(n!)$ (Fatorial, crescimento muito rápido)
3. $O(2^n)$ (Exponencial, crescimento rápido)
4. $O(n^2)$ (Quadrática)
5. $O(n \log n)$ (Linearitmica)
6. $O(n)$ (Linear)
7. $O(\log n)$ (Logarítmica)
8. $O(1)$ (Constante, a mais eficiente)

Questão 2.

- a) $T(n)=T(n-1)+c$
- b) $T(n)=T(n-1)+c$

Questão 3.

O (Big O)

A notação Big O indica o pior caso do crescimento de um algoritmo. Ela mostra o limite superior da taxa de crescimento, ou seja, o quanto rápido o tempo de execução pode crescer no máximo.

Exemplo: Se um algoritmo tem tempo $O(n^2)$, isso significa que o tempo de execução não cresce mais rápido do que uma função quadrática, mesmo que às vezes seja mais rápido.

Ω (Ômega)

A notação Ω indica o melhor caso, mostrando o limite inferior do crescimento. Ela descreve a velocidade mínima com que o tempo de execução pode crescer.

Exemplo: Se um algoritmo é $\Omega(n)$, significa que ele pelo menos cresce linearmente, não pode ser mais rápido do que isso em termos de ordem de grandeza.

Θ (Teta)

A notação Θ é usada quando podemos determinar tanto o limite superior quanto o inferior da mesma ordem de grandeza. Ou seja, o algoritmo sempre cresce de forma proporcional a uma determinada função.

Exemplo: Se um algoritmo é $\Theta(n \log n)$ isso quer dizer que o tempo de execução cresce exatamente na ordem de $n \log n$, nem mais rápido nem mais lento, em termos assintóticos.

Questão 4.

Melhor caso

- Ocorre quando $v[n-1] == x$ na primeira verificação (elemento está na posição $n-1$).
- Comparações feitas: constante.
- Complexidade: $\Theta(1)$ (também $O(1)$ e $\Omega(1)$).

Pior caso

- Ocorre quando x não está no vetor ou está na posição 0 — a função precisa checar todos os elementos até $n = 0$.
- Comparações feitas: aproximadamente n .
- Complexidade: $\Theta(n)$ (também $O(n)$ e $\Omega(n)\Omega(n)$).

Caso médio

- Se assumirmos que x está presente em uma posição aleatória uniformemente entre 0 e $n-1$, o número esperado de comparações é $(n+1)/2(n+1)/2(n+1)/2$, ou seja, ordem linear.
- Complexidade média: $\Theta(n)$.

Questão 5.

- $\log \sqrt{n} = O(\log n)$. Verdadeiro.
 $\log \sqrt{n} = (\frac{1}{2})\log(n) = C * \log(n)$, portanto $O(\log n)$
- Verdadeiro, pela transitividade, $f = \Theta(h)$
- Falso, contra-exemplo: $f(n)=1$, $g(n)=n$, $h(n)=n \Leftrightarrow f=O(g)$, $g=\Theta(h)$, mas $f \neq \Theta(h)$
- se $f = O(g)$ então $2f = O(2g)$. Verdadeiro. Ao multiplicar ambos por 2 a relação não é alterada, pois constantes não importam para $O()$.

RECURSIVIDADE

Questão 1.

A complexidade da função será $O(n)$, pois ela é chamada uma vez para cada valor de n até chegar a 0, ou seja, será chamada n vezes.