Lista de exercícios

Noções de complexidade de algoritmos

1. Quais critérios podemos utilizar para escolher entre dois algoritmos diferentes que resolvem o mesmo problema?

R: A simplicidade e o consumo de recursos.

2. O que é complexidade de algoritmo?

R: A complexidade de algoritmo é um modelo matemático que mede os recursos que um algoritmo consome, geralmente em termos de tempo e memória, conforme o tamanho de entrada aumenta.

3. O que é análise de algoritmo?

R: É o processo de determinar a complexidade de um algoritmo ou é o estudo da eficiência em termos de recursos computacionais.

4. Quais são as duas principais formas de análise de algoritmos?

R: Experimental e teórica

Experimental: baseada na implementação prática e testes empíricos.
Teórica: baseada em uma análise matemática, geralmente usando a notação Big-O.

5. Qual é o propósito da notação assintótica?

R: É o processo de avaliar o desempenho de um algoritmo, tanto em termos de tempo de execução quanto de uso de memória, à medida que o tamanho da entrada (n) varia.

6. O que significa dizer que uma função f(n) é O(g(n))?

R: Quer dizer que **f(n)** cresce no máximo na mesma taxa que **g(n)**

7. Considere o seguinte algoritmo que verifica que calcula a amplitude dos valores de uma lista não vazia
    a. Para uma lista com n elementos, quantas vezes as operações < e > são executadas? Qual é a complexidade de tempo do algoritmo?

R: Total de operações = (n - 1) + (n- 1) = 2(n - 1) = 2n - 2 .

A complexidade de tempo do algoritmo é O(n), já que a execução do algoritmo aumenta linearmente com o número de elementos da lista.

8. Considere o seguinte algoritmo que verifica se uma lista é palíndromo
    a. Para uma lista de tamanho n, quantas vezes no mínimo a operação != será executada? Como deve estar a entrada para que isso aconteça? Qual é a complexidade de tempo do algoritmo no melhor caso?

R: O mínimo de operações = 1.

A entrada deve ser, quando o primeiro elemento e o último elemento da lista são diferentes.

A complexidade de tempo do algoritmo é O(1). Pois o algoritmo termina após a primeira comparação.

    b. Para uma lista de tamanho n, quantas vezes no máximo a operação != será executada? Como deve estar a entrada para que isso aconteça? Qual é a complexidade de tempo do algoritmo no pior caso?

R: O máximo de operações = n / 2.

A entrada deverá ser um palíndromo.

A complexidade de tempo do algoritmo é O(n). Pois o algoritmo percorre metade da lista.

9. O algoritmo de ordenação por inserção usa o método incremental para ordenar uma lista de valores. Quando o algoritmo está processando o elemento da posição i, a sublista de 0 até i−1 já está ordenada, então o trabalho do algoritmo é mover o elemento da posição i de maneira a sublista de 0 até que fique ordenada. A seguinte função implementa o algoritmo de ordenação por inserção.
    a. Para uma lista de tamanho n, quantas vezes no mínimo as comparações da linha do while serão executadas? Como deve estar a entrada para que isso aconteça? Qual é a complexidade de tempo do algoritmo no melhor caso?

R: O mínimo de operações = n - 1.

A entrada deverá ser uma lista já ordenada.

A complexidade de tempo do algoritmo é O(n).

b. Para uma lista de tamanho n, quantas vezes no máximo as comparações da linha do while serão executadas? Como deve estar a entrada para que isso aconteça? Qual é a complexidade de tempo do algoritmo no pior caso?

R: O máximo de operações = $n(n - 1) / 2 = n^2 - n / 2$.

A entrada deverá estar em ordem decrescente.

A complexidade de tempo do algoritmo é $O(n^2)$.