Análise e Técnicas de Algoritmos
Questões de Provas de Períodos Anteriores

1. Em sala de aula, estudamos e analisamos o Quicksort. Para tanto, apresentamos um procedimento de partição que definia o último elemento do array como pivô. Vamos definir um Quicksort diferente, onde o pivô é o primeiro elemento do array.
    a. Escreva, em pseudo-código, um procedimento que efetua a partição do array para este novo Quicksort.
    b. Ilustre a operação de partição, considerando o array A=[13, 19, 9, 5, 12, 8, 7, 4, 11, 2].
    c. Faça a análise de complexidade do novo Quicksort para o melhor caso, pior caso e para o caso médio.

2. Considere um array A de números reais.
    a. Escreva um algoritmo O(n.log n) que encontra um par de elementos (x, y) de A, em que |x + y| é mínimo.

3. Argumente que o seu algoritmo está correto e efetue a análise de complexidade de sua solução. Considere o problema de selos de postagem.
    a. Mostre que qualquer postagem de valor inteiro maior do que 7 centavos pode ser formada utilizando apenas selos de 3 e 5 centavos.
    b. Escreva um algoritmo que dado um valor da postagem retorna o número de selos de 3 e 5 centavos, respectivamente.
    c. Mostre que o algoritmo está correto.
    d. Indique a complexidade do algoritmo.

4. Um problema que está relacionado com o problema de ordenação é o de encontrar o *k*-ésimo menor elemento de uma lista não ordenada.
    a. Escreva um algoritmo que encontra o k-ésimo menor elemento. Esse algoritmo deve ser $\Theta(n)$ na média e no melhor caso.
    b. Argumente que seu algoritmo está correto.
    c. Efetue a análise de complexidade de sua solução para melhor caso, pior caso e caso médio.
5. Vimos em sala de aula que o *Counting Sort* é um algoritmo de ordenação **estável**. O que dizer dos algoritmos baseados em comparação que estudamos: *InsertionSort*, *MergeSort* e *QuickSort*? Justifique a sua resposta.

6. Considere um array que consiste de ***n*** elementos. Cada elemento pode ser *vermelho*, *branco* ou *azul*. Ordenar os elementos de tal forma que todos os vermelhos apareçam antes de todos os brancos e, por sua vez, todos os brancos venham antes de todos os elementos de cor azul. Além do caminhamento no array, as únicas operações permitidas são:

    - **Examina(*i*)**: avalia a cor do ***i***-ésimo elemento no array.

- **Swap(*i, j*)**: troca o ***i***-ésimo elemento com o ***j***-ésimo elemento do array.

    a. Escreva, em pseudo-código, um algorimo de tempo linear que efetua a ordenação vermelho-branco-azul.
    b. Se agora considerarmos ***k*** cores, explique como modificar o algoritmo do item (a) para que a ordenação das ***k*** cores seja efetuada em tempo ***O(n logk)***.

7. O Professor de ATAL bolou um algoritmo para fazer o merge de ***k*** listas ordenadas, cada uma com ***n/k*** elementos. O algoritmo pega a primeira lista e faz o merge com a segunda lista, utilizando um algoritmo de tempo linear para fazer o merge de duas listas ordenadas, como o algoritmo de merge do MergeSort. Então, um merge é feito com a lista resultante de ***2n/k*** elementos e a terceira lista, um novo merge com a lista resultante de ***3n/k*** elementos e a quarta lista, e assim por diante, até conseguir uma única lista ordenada com todos os ***n*** elementos. Faça uma análise de pior caso do tempo de execução do algoritmo do professor de ATAL. Apresentar o resultado em termos de ***n*** e ***k***.

8. Considere ***n*** pontos distribuídos ao longo do eixo dos **x**. O objetivo é encontrar os dois pontos com menor distância mútua.
    a. Escreva um algoritmo baseado em divisão-e-conquista que resolve o problema. Indique a complexidade do algoritmo.
    b. Se utilizarmos a solução ingênua, qual a complexidade desta solução?

9. Seja um conjunto de ***n*** garrafas distintas e *n* correspondentes rolhas (existe uma correspondência de um-pra-um entre garrafas e rolhas). Não é permitido comparar duas garrafas ou duas rolhas diretamente, mas é permitido comparar uma garrafa e uma rolha para saber quem é maior das duas. Escreva um algoritmo para encontrar o casamento entre garrafas e rolhas. É esperado que, no caso médio, a complexidade de seu algoritmo seja ***O(n.log n)***.

10. Considere uma lista com ***n*** números distintos $x_1, x_2, ..., x_n$, em ordem arbitrária, e um valor $k < n$. Escreva em pseudo-código, um algoritmo que imprime os ***k*** menores valores da lista, em qualquer ordem. Por exemplo, se $k = 3$, deve ser impresso os ***3*** menores valores de $x_1, x_2, ..., x_n$. O algoritmo deve resolver o problema em ***O(n)***, independente do tamanho de ***k***.

11. Considere um array ***A***, contendo ***n*** números distintos. Esse array tem a seguinte propriedade: existe um índice ***m*** de modo que os números do subarray ***A[1..m]*** estão em ordem crescente e os números do subarray ***A[m+1..n]*** estão em ordem decrescente. Escreva (em pseudo-código) um algoritmo ***O(log n)*** que retorna o

índice ***m***. Por exemplo, se o array de entrada é ***A[1..9] = [3; 7; 8; 9; 6; 5; 4; 2; 1]***, a saída deve ser ***m = 4***. Argumente que o seu algoritmo está correto.