SSC0503 - Introdução à Ciência de Computação II
4ª Lista
Professor: Claudio Fabiano Motta Toledo (`claudio@icmc.usp.br`)
Estagiário PAE: Jesimar da Silva Arantes (`jesimar.arantes@usp.br`)

---

1. Desenvolva manualmente o algoritmo insertion sort sobre o arranjo $A = [13, 19, 9, 5, 12, 8, 7, 4]$.
2. Desenvolva manualmente o algoritmo bubble sort sobre o arranjo $A = [13, 19, 9, 5, 12, 8, 7, 4]$.
3. Desenvolva manualmente o algoritmo selection sort sobre o arranjo $A = [13, 19, 9, 5, 12, 8, 7, 4]$.
4. Desenvolva manualmente o algoritmo quicksort sobre o arranjo $A = [13, 19, 9, 5, 12, 8, 7, 4, 11, 2, 6, 21]$.
   - Pegue o primeiro elemento como pivo.
   - Pegue o elemento do meio como pivo.
   - Pegue um elemento aleatório como pivo.
   - Pegue a mediana de três como pivo.
5. Desenvolva manualmente o algoritmo quicksort com pivo escolhido aleatoriamente sobre os arranjos:
   - $A = [0, 1, 3, 4, 5, 7, 9]$
   - $A = [1, 0, 4, 3, 5, 9, 7]$
   - $A = [1, 2, 3, 4, 3, 2, 1]$
6. Use o método de substituição para provar que a recorrência $T(n) = T(n-1) + \Theta(n)$ tem a solução $T(n) = \Theta(n^2)$.
7. Use o método mestre para provar que a recorrência $T(n) = 2T(n/2) + \Theta(n)$ tem a solução $T(n) = \Theta(n \cdot lg\ n)$.
8. Qual é o tempo de execução de quick sort quando todos os elementos do arranjo A têm o mesmo valor?
9. Desenvolva um programa em C que faça a ordenação através do método quicksort sobre um vetor de tamanho N. Em seguida, diga qual a análise de complexidade no melhor caso, pior caso e caso médio.
10. Desenvolva um programa em C que faça a ordenação (em ordem decrescente) através do método quicksort sobre um vetor de tamanho N. Em seguida, diga qual a análise de complexidade no melhor caso, pior caso e caso médio.
11. Suponha que as divisões em todo nível do quicksort são na proporção $1 - \alpha$ para α, onde $0 \leq \alpha \leq 1/2$ é uma constante. Mostre que a profundidade mínima de uma folha na árvore de recursão é aproximadamente $-lg\ n/lg\ \alpha$ e a profundidade máxima é aproximadamente $-lg\ n/lg\ (1 - \alpha)$. Não se preocupe com arredondamentos.

12. Mostre que para $T(n) = max_{0 \leq q \leq n-1}(T(q) + T(n-q-1)) + \Theta(n)$ temos $T(n) = \Omega(n^2)$.

13. Mostre que o quicksort no melhor caso executa em $\Omega(n \cdot lg\ n)$.

14. No algoritmo Randomized-Quicksort, quantas chamadas são feitas ao gerador de números aleatórios Random no pior caso? e no melhor caso?. Dê sua resposta em termos da notação Θ.

15. Desenvolva um programa em C que faça uma implementação combinada do quicksort com o insertion sort. O quicksort será executado até que o partição fique menor que um determinado valor k (por exemplo, $k = 10$, $k = 20$) então faça uma chamada para o método insertion sort. Avalie e compare o desempenho isolado do quicksort e insertion sort.

16. Pratique a execução do algoritmo heapsort no vetores dos exercícios 1,2,3 e 5.

17. Quais são os números mínimo e máximo de elementos numa heap de altura h?

18. Mostre que uma heap com n elementos tem altura $\lfloor lg\ n \rfloor$.

19. Mostre que, armazenando uma heap com n elementos em um vetor, os nós folhas são os nós indexados por $\lfloor n/2 \rfloor + 1, \lfloor n/2 \rfloor + 2, \cdots, n$.

20. Escreva um algoritmo para a rotina Min-Heapify(A,i). Compare o tempo de execução da Min-Heapify com Max-Heapify.

21. Qual o efeito de chamar Max-Heapify(A,i), quando o elemento A[i] é maior que seus filhos? e quando i>A.heap-size/2?

22. O código para Max-Heapify é bastante eficiente, exceto pela chamada recursiva na linha 10. Escreva uma rotina Max-Heapify que usa um controle iterativo ao invés de recursão.

23. Mostre que há no máximo $\lceil n/2^{h+1} \rceil$ nós de altura h em qualquer heap com n elementos.

24. Mostre que o pior caso no tempo de execução do algoritmo heapsort é $\Omega(n \cdot lg\ n)$.

25. Execute Max-Heap_Insert(A, 10) na heap $A = [15, 13, 9, 5, 12, 8, 7, 4, 0, 6, 2, 1]$.

26. Escreva algoritmos para Heap-Minimum, Heap-Extract-Min, Heap-Decrease-Key e Min-Heap-Insert que implemente uma min-priority queue com uma min-heap.

27. Cada operação de troca (exchange) na linha 5 da Heap-Increase-Key requer tipicamente três atribuições. Mostre como usar a ideia de laço interno (inner loop) do Insertion-Sort para reduzir as três atribuições para apenas uma atribuição.

28. Mostre como implementar uma fila first-in, first-out com priority queue. Mostre como implementar uma pilha com priority queue.