Lista de Algoritmos avançados:

1. (Exercício 2.1-3 do livro do Cormen) Escreva um pseudocódigo para a busca linear e mostre, usando invariância de laço, que o seu algoritmo está correto.

2. Implemente o algoritmo de ordenação por inserção e crie uma cópia anotada dele que mede o número de operações no modelo da Random Access Machine (RAM, seção 2.2 livro do Cormen). Usando entradas de tamanho crescente, mostre em um gráfico quando o tempo de execução no modelo RAM diverge de medições feitas em uma máquina real.

3. Mostre numericamente com suas implementações dos algoritmos de insertion-sort e merge-sort como se comporta o desempenho de cada algoritmo utilizando entradas de tamanho crescente, considerando entradas de pior caso, melhor caso e caso médio. Análise, para cada tipo de entrada, se existe algum ponto a partir do qual um algoritmo passa a ser mais rápido que o outro.

4. Mostre numericamente com suas implementações dos algoritmos de multiplicação de matrizes que o algoritmo de Strassen é mais rápido que o algoritmo convencional.

5. Escolha um algoritmo recorrente para aplicar um dos 4 métodos de resolução de recorrência descritos no capítulo 4 para medir o custo da recorrência do algoritmo escolhido. Compare o resultado com medições de tempo.

6. O problema de balanceamento de cargas busca atribuir tarefas de tamanhos diferentes a trabalhadores, de modo a minimizar a carga máxima que um trabalhador irá executar. Em um problema em que temos n tarefas e k trabalhadores (n > k), considere que o balanceador irá distribuir as n/k primeiras tarefas para o primeiro trabalhador, as n/k tarefas seguintes para o segundo trabalhador, e assim por diante. Mostre numericamente como permutar aleatoriamente os dados de entrada, que são as cargas de cada tarefa, pode influenciar na solução desse balanceador.
7. Implemente as funções da seção 6.5 (Priority queues) do livro do Cormen 4th Ed. em sua linguagem favorita e proponha um exemplo de uso com uma demonstração.

8. Mostre com experimentos numéricos quando suas próprias implementações de Quicksort e do Quicksort aleatório são mais vantajosas, comparando uma com a outra.

9. Mostre com experimentos numéricos quando o Radix-sort com o Count-sort é mais rápido que o Count-sort sozinho. Utilize suas próprias implementações ou alguma implementação existente explicando os resultados.

10. Implemente o algoritmo de mínimo e máximo simultâneos da seção 9.1 do livro do Cormen, 4ª Ed., na sua linguagem favorita e mostre através de medição de tempo que é mais rápido que a abordagem não-simultânea para um vetor de entrada suficientemente grande.
11. Implemente os algoritmos de seleção aleatória e seleção das seções 9.2 e 9.3 do livro do Cormen, 4ª Ed., e realize experimentos numéricos para demonstrar em quais casos um tem vantagens com relação ao outro.

12. Implemente o algoritmo da mediana ponderada e use-o para resolver o item e do Problema 9-3 do Cormen, 4ª Ed.
13. Implemente as estruturas de dados: 1) array ordenado, 2) lista encadeada não ordenada, e 3) árvore de busca binária balanceada (e.g., AVL, Red-Black trees). Em seguida, insira n elementos em cada estrutura de dados. Depois disso, considerando diferentes valores de n, compare os tempos de inserção e busca de cada uma.
14. Implemente tabelas hash com encadeamento e usando endereçamento aberto. Realize experimentos para mostrar numericamente as vantagens e desvantagens de cada caso, considerando as operações de inserção, busca e remoção.
15. Compare implementações de tabelas hash com hashing duplo e probing linear. Analise e justifique os resultados dos seus experimentos.

16. Defina um grafo acíclico direcionado relacionado a área do seu mestrado/doutorado e use a sua implementação para ordenar topologicamente esse grafo.
17. Implemente e aponte vantagens e desvantagens dos algoritmos de Kruskal e Prim para gerar uma árvore de abrangência mínima. Use exemplos práticos da sua área para demonstrar suas conclusões.
18. Implemente o algoritmo do Dijkstra e utilize-o para resolver um problema prático da sua área de interesse.
19. Estude e apresente como as primitivas de paralelismo spawn, sync, and parallel for podem ser relacionadas com o padrão e modelo de programação OpenMP.
20. Escolha um dos algoritmos que já implementou nas listas anteriores que poderiam se beneficiar de paralelismo e implemente-os utilizando OpenMP.
21. Apresente uma análise experimental do algoritmo implementado na questão 2 para realizar suas medições (sugestão: utilizar o NPAD).
22. Mostre experimentalmente que a taxa de competitividade do algoritmo de mover para a frente se aproxima de 4 estatisticamente.
23. Mostre experimentalmente que a taxa de competitividade do algoritmo de uso menos recente tem taxa de competitividade independentemente do tamanho da entrada para o mecanismo de memória cache.
24. Compare o desempenho do algoritmo de marcação aleatória com o algoritmo de uso menos recente para qualquer um dos programas que já implementou na disciplina. Dica: extraia o traço de memória do seu programa e use-o como entrada para a sua comparação.
25. Formule um problema de programação linear e desenvolva um programa para obter a matriz do tableau;
26. Desenvolva um programa para obter a solução ótima do problema primal a partir do tableau;
27. Obtenha o problema dual associado e a matriz do tableau do problema dual. Ache a solução ótima do problema dual com o tableau. Em seguida, encontre a solução do problema primal a partir da solução do problema dual.
28 .Implemente os algoritmos de resolução de sistemas lineares por substituição direta e reversa;
29. Implemente o algoritmo de decomposição LUP
30. Use os algoritmos que implementou nessa lista e crie uma rotina para calcular a inversa de uma matriz nxn não-singular qualquer, verificando o resultado com o auxílio de algum algoritmo de multiplicação de matrizes que já implementou.
31. Implemente um algoritmo para computar a DFT de um polinômio qualquer.

32. Implemente o algoritmo da FFT e compare seu desempenho com o da DFT para diferentes tamanhos de problemas.
33. Escreva um programa que mostre que sua implementação da FFT está correta. Para isso, use a dualidade entre multiplicação e convolução nas representações de polinômios usando coeficientes e valor-ponto.