Questionário Algoritmos de Ordenação

Total de pontos 100/100

O e-mail do participante (**martinson.freitas@gmail.com**) foi registrado durante o envio deste formulário.

✓ 1. Qual das seguintes é uma das principais razões para a inclusão de algoritmos de ordenação em um livro sobre estruturas de dados, de acordo com o texto? *10/10

- ○ Porque todas as estruturas de dados podem ser ordenadas em tempo constante.
- ◉ Devido à íntima relação de dois algoritmos de classificação (quicksort e heap-sort) com estruturas de dados como árvores aleatórias de busca binárias e pilhas (heaps), respectivamente. ✓
- ○ Para demonstrar que estruturas de dados complexas não são necessárias para a ordenação.
- ○ Eles são os únicos algoritmos que utilizam o conceito de recursão.
- ○ Apenas para preencher o capítulo, já que a ordenação não tem relação direta com estruturas de dados.

✓ 2. Qual é o limite inferior assintótico para o número de comparações que *qualquer* algoritmo de ordenação baseado em comparações (determinístico ou randomizado) deve realizar no pior caso e até mesmo no caso médio, para classificar n itens? *10/10

- ○ O(n)
- ○ O(n^2)
- ○ O(log n)
- ◉ O(n log n). ✓
- ○ O(n!)

✓ 3. Em contraste com os algoritmos de ordenação baseados em comparação (como **Merge-sort**, **Quicksort** e **Heap-sort**), qual tipo de operação os algoritmos não-comparativos (como **Counting Sort** e **Radix Sort**) utilizam para potencialmente alcançar uma classificação mais rápida, evitando os limites inferiores de tempo? *10/10

- ○ Trocas de elementos adjacentes de forma iterativa.
- ○ Recorrência complexa e uso intensivo de chamadas de função.
- ◉ Indexação de array, utilizando partes dos elementos como índices. ✓
- ○ Randomização para otimização do caso médio.
- ○ Redução de problemas a subproblemas de tamanho um.

✓ 4. O algoritmo **Merge-sort** é um exemplo clássico de qual paradigma de projeto de algoritmos, onde um problema é dividido em subproblemas menores, resolvidos recursivamente e então combinados? *10/10

- ○ Força Bruta.
- ○ Programação Dinâmica.
- ○ Backtracking.
- ◉ Divisão e conquista recursiva. ✓
- ○ Algoritmo Guloso.

✓ 5. Qual é o número máximo de comparações que o algoritmo **Merge-sort** executa para ordenar n elementos, e qual é a sua complexidade de tempo de execução no pior caso? *10/10

- ○ O(n) comparações, tempo O(n).
- ○ O(n^2) comparações, tempo O(n^2).
- ○ No máximo 2n log n comparações, tempo O(n log n).
- ○ No mínimo log(n!) comparações, tempo O(n).
- ◉ No máximo n log n comparações, tempo O(n log n). ✓

✓ 6. O **Quicksort** é um algoritmo de ordenação que se relaciona intimamente com qual estrutura de dados, especialmente na sua versão randomizada? *10/10

- ○ Árvores B.
- ○ Árvores AVL.
- ◉ Árvores de pesquisa binárias aleatórias. ✓
- ○ Heaps Binárias.
- ○ Listas Skip.

✓ 7. Qual das seguintes afirmações é *verdadeira* sobre o desempenho esperado do **Quicksort** quando o array de entrada contém elementos duplicados? *10/10

- ○ O Quicksort não pode classificar arrays com elementos duplicados.
- ○ O tempo de execução esperado do Quicksort é O(n^2) se houver muitos duplicados.
- ○ A presença de duplicados não afeta em nada o desempenho.
- ○ Elementos duplicados devem ser removidos antes de chamar o Quicksort.
- ◉ O tempo de execução esperado do Quicksort não é pior e pode ser ainda melhor, pois todas as ocorrências de um pivô duplicado são agrupadas e não participam dos subproblemas recursivos. ✓

✓ 8. Qual é a complexidade de tempo para a etapa de construção inicial de um **heap** (transformar um array não ordenado de *n* elementos em um **BinaryHeap**) no algoritmo **Heap-sort**, usando a estratégia de baixo para cima? *10/10

- ○ O(n log n), pois envolve n chamadas de add(x).
- ○ O(n log n), porque cada elemento deve ser comparado com todos os outros.
- ○ O(log n), pois é uma operação em heap.
- ○ O(1), pois é uma operação simples de alocação de memória.
- ◉ O(n), que é mais eficiente do que chamar add(x) n vezes. ✓

✓ 9. Qual das seguintes características é uma notável do **Merge-sort** em comparação com o **Quicksort** e o **Heap-sort**, conforme discutido no livro? *10/10

- ○ É um algoritmo randomizado, o que significa que seu tempo de execução não é garantido.
- ○ Tem o maior número de comparações entre os algoritmos O(n log n).
- ○ É determinístico e no local.
- ◉ Ele usa um array auxiliar durante a fase de mesclagem, o que pode ser caro e um ponto potencial de falha se a memória for limitada. ✓
- ○ Sua complexidade de tempo de execução esperada é O(n^2).

✓ 10. O algoritmo **Counting Sort** é descrito como "*estável*". O que significa essa propriedade para a ordenação de elementos? *10/10

- ○ Ele sempre classifica os elementos na mesma ordem, independentemente da entrada.
- ○ Ele tem uma complexidade de tempo de O(1), tornando-o muito rápido.
- ○ Significa que o algoritmo nunca falhará ou produzirá um erro.
- ◉ Ele preserva a ordem relativa de elementos iguais: se dois elementos têm o mesmo valor e um aparece antes do outro na entrada, ele aparecerá antes do outro na saída. ✓
- ○ Indica que o algoritmo pode ser interrompido a qualquer momento sem perda de dados.