1. Busca Binária
   1. Sua complexidade é O(log n), o que significa que o tempo de execução cresce de forma logarítmica em relação ao tamanho da entrada.
2. Bubble Sort
   1. A complexidade de tempo do Bubble Sort é O(n^2), o que significa que o tempo de execução cresce quadratically em relação ao tamanho da entrada.
3. Fatorial
   1. A complexidade desse algoritmo é O(n), pois o número de operações cresce linearmente com o tamanho da entrada (n).
4. QuickSort
   1. A complexidade de tempo média do QuickSort é O(n log n), tornando-o eficaz para ordenar grandes conjuntos de dados.
5. Linear Search
   1. A complexidade de tempo do algoritmo de busca linear é O(n), o que significa que o tempo de execução cresce linearmente com o tamanho da entrada.
6. Merge Sort
   1. A complexidade de tempo do Merge Sort é O(n log n), tornando-o eficaz para ordenar grandes conjuntos de dados.
7. Seleção de Elemento Mínimo
   1. Sua complexidade é O(n^2), o que significa que o tempo de execução cresce de forma quadrática em relação ao tamanho da entrada.
8. Multiplicação de Matrizes
   1. A multiplicação de matrizes é uma operação com complexidade O(n^3), uma vez que envolve três loops aninhados para calcular cada elemento da matriz resultante.
9. Busca em Árvore Binárias
   1. Em uma árvore binária balanceada, a complexidade média é O(log n), onde "n" é o número de nós na árvore. No entanto, em uma árvore binária não balanceada, a complexidade pode ser O(n) no pior caso, se a árvore estiver completamente desbalanceada.
10. Hashing
    1. A complexidade da operação de inserção e busca em uma tabela de dispersão (hash table) é, em média, O(1) quando a função de dispersão é bem projetada