Existem muitos tipos de algoritmos de ordenação, cada um com suas vantagens e desvantagens. Algumas das opções mais comuns incluem Bubble Sort, Insertion Sort, Selection Sort, Merge Sort, Quick Sort, Heap Sort, e também Shell Sort, Radix Sort, Counting Sort e Bucket Sort.

Algoritmos de Ordenação:

- **Bubble Sort:**

Um dos algoritmos mais simples, compara elementos adjacentes e os troca se estiverem fora de ordem. É ineficiente para grandes conjuntos de dados.

- **Insertion Sort:**

Itera sobre a lista, inserindo cada elemento na sua posição correta no subconjunto já ordenado. É eficiente para listas pequenas ou quase ordenadas.

- **Selection Sort:**

Encontra o menor elemento da lista e o troca com o primeiro elemento. Repete o processo para o restante da lista. É simples, mas ineficiente para grandes conjuntos de dados.

- **Merge Sort:**

Divide a lista em sublistas, ordena cada uma e depois as mescla. É um algoritmo eficiente, com complexidade O(n log n).

- **Quick Sort:**

Escolhe um "pivô" e divide a lista em duas partes: elementos menores e maiores que o pivô. É geralmente muito rápido, mas pode ter um desempenho ruim em certos casos.

- **Heap Sort:**

Utiliza uma estrutura de dados chamada heap para ordenar a lista. É eficiente e tem uma complexidade O(n log n).

- **Shell Sort:**

Um algoritmo de ordenação de shell, que usa um intervalo decrescente para comparar e trocar elementos. É mais eficiente que o Insertion Sort e Shell Sort.

- **Radix Sort:**

Ordena elementos dígito por dígito. É eficiente para números inteiros.

- **Counting Sort:**

Ordena elementos contando a frequência de cada elemento. É eficiente para conjuntos de dados com um intervalo limitado de valores.

- **Bucket Sort:**

Divide a lista em "buckets" e ordena cada bucket separadamente. É eficiente para dados com uma distribuição uniforme.