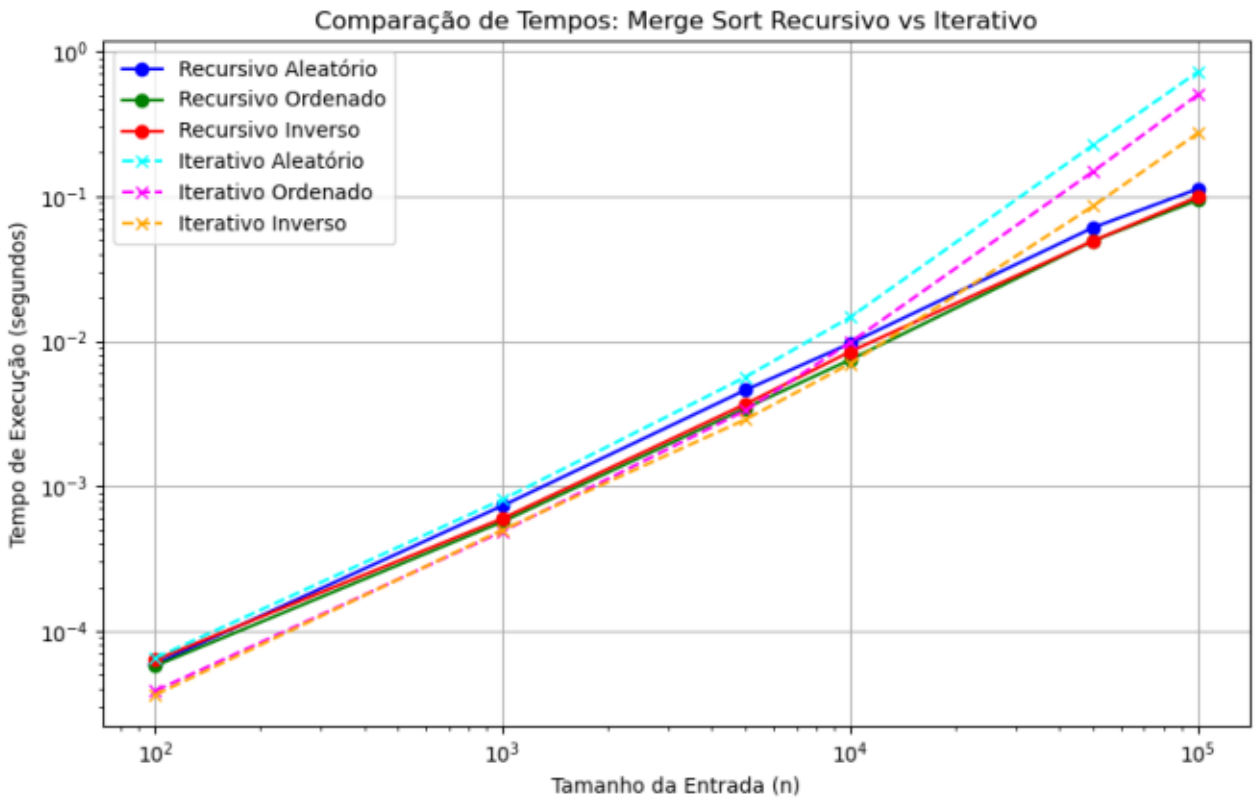
Comparação de Tempos: Merge Sort Recursivo vs Iterativo



Comparação da Implementação Recursiva x Iterativa do Merge Sort

Após implementar e testar o algoritmo Merge Sort nas versões recursiva e iterativa, foram observadas as seguintes considerações:

1. Clareza do Código:
- A implementação recursiva é geralmente mais clara e concisa, refletindo a lógica do algoritmo de maneira mais direta. A recursão é mais fácil de entender para aqueles que estão familiarizados com a divisão e conquista.
- A versão iterativa, embora funcional, tende a ser um pouco mais complexa e difícil de seguir, especialmente devido à necessidade de gerenciar manualmente a mesclagem das sublistas.

2. Desempenho:
- Em termos de tempo de execução, ambas as implementações têm complexidade teórica O(n log n). Contudo, a implementação recursiva pode ser mais lenta em alguns casos devido ao overhead das chamadas de função e à profundidade da pilha, especialmente para n menores.
- A implementação iterativa, por outro lado, tende a ser mais eficiente em termos de uso de memória, pois não utiliza a pilha de chamadas recursivas, o que pode levar a estouros de pilha em listas muito grandes, especialmente para n menores.
- Por outro lado, para n a partir de valores 10 ^ 4, a versão recursiva é mais eficiente devido a otimizações como realizar a mesclagem em sublistas que são geralmente menores e, portanto, mais eficientes. Isso ocorre porque em listas maiores, as vantagens de dividir a lista em sublistas e resolver cada parte de forma independente tornam-se mais evidentes na versão recursiva.

3. Uso de Memória:
- A implementação recursiva pode consumir mais memória, especialmente para entradas grandes, devido à pilha de chamadas. Cada chamada recursiva consome espaço na pilha, o que pode levar a problemas em entradas grandes.
- A versão iterativa, em contrapartida, gerencia a mesclagem dentro de um único bloco de memória, o que a torna mais robusta para listas grandes.

Essas observações sublinham a importância de considerar tanto a clareza do código quanto o desempenho e o uso de memória ao escolher entre abordagens recursivas e iterativas em algoritmos de ordenação.

Resultados de Tempo de Execução - Recursivo

Tamanho da Entrada (n)	Tempo Recursivo Aleatório (s)	Tempo Recursivo Ordenado (s)	Tempo Recursivo Inverso (s)
100.0	5.91278076171875e-05	5.745887756347656e-05	6.318092346191406e-05
1000.0	0.0007340908050537109	0.0005688667297363281	0.0005986690521240234
5000.0	0.0046138763427734375	0.0034584999084472656	0.0036902427673339844
10000.0	0.009668111801147461	0.00746607780456543	0.00848579406738281 2
50000.0	0.060941457748413086	0.04928302764892578	0.04938006401062012
100000.0	0.11267375946044922	0.09430694580078125	0.09909796714782715

Resultados de Tempo de Execução - Iterativo

Tamanho da Entrada (n)	Tempo Iterativo Aleatório (s)	Tempo Iterativo Ordenado (s)	Tempo Iterativo Inverso (s)
100.0	6.508827209472656e-05	3.8623809814453125e-05	3.62396240234375e-05
1000.0	0.0008096694946289062	0.0004880428314208984	0.0004999637603759766
5000.0	0.0056705474853515625	0.003339529037475586	0.002891063690185547
10000.0	0.014660120010375977	0.009818315505981445	0.007014274597167969
50000.0	0.22628092765808105	0.14872527122497559	0.08585405349731445
100000.0	0.7221317291259766	0.5033233165740967	0.2728271484375