João Felipe Veras Dantas Rocha / DRE:117202753
Lucas dos Santos Melo / DRE:123312281
Rafael Augusto Santos / DRE:124155331
Rayan Lucas de Paula Carvalhal / DRE:124180742
Link para o GitHub dos arquivos com os códigos, assim como o arquivo de texto: https://github.com/jfvdrocha/BD-2025.1-ordenacao-externa-por-mergesort

# Ordenação Externa por MergeSort

Professor: Milton Ramos Ramirez

Rio de Janeiro

2025

## Comentários sobre o Desenvolvimento

O projeto de ordenação externa com Merge Sort teve como objetivo organizar grandes arquivos CSV que excedem a capacidade da memória principal. A implementação baseou-se no clássico paradigma de divisão e conquista, adaptando o algoritmo Merge Sort para operar sobre dados em memória secundária.

### Estruturas de Dados Utilizadas:

- Listas: Utilizadas para armazenar temporariamente os dados de cada run durante a leitura parcial dos arquivos.
- Min-heap (via heapq): Utilizado na fase de merge externo para realizar a intercalação eficiente dos menores elementos entre múltiplos arquivos temporários ordenados.
- Arquivos temporários: Criados usando o módulo tempfile, representam os runs ordenados. Permitem simular um ambiente de limitação de RAM, gerenciando os dados pela memória secundária (disco).

### Divisão de Módulos e Funções

A solução foi modularizada de forma clara e pedagógica:

- merge_sort(): Implementa o algoritmo clássico de Merge Sort para ordenação em memória principal.
- intercalar(): Função auxiliar para realizar a intercalação de duas listas ordenadas.
- criar_runs_ordenados(): Divide o arquivo CSV em blocos (runs) que cabem na RAM, ordena cada bloco com Merge Sort e salva em arquivos temporários.
- salvar_run_temporario(): Função auxiliar que salva uma lista ordenada de registros em um arquivo temporário.
- merge_externo(): Realiza a intercalação final dos arquivos runs ordenados, gerando o arquivo CSV final.
- inverter_chave(): Permite ordenação decrescente ao adaptar os valores das chaves para uso no heap.
- ordenacao_externa(): Função orquestradora que gerencia toda a execução da ordenação externa.

### Complexidade de Tempo e Espaço:

Tempo:

- Fase de ordenação de runs: O tempo é proporcional a O(N log N) no total, pois cada run (subconjunto do arquivo) é ordenado individualmente.
- Fase de merge externo: O uso de um heap permite que a intercalação entre k arquivos seja feita em tempo O(N log k), onde k é o número de runs.

Espaço:

- RAM: Limitado ao tamanho do buffer configurado (buffer_max_linhas), controlando o consumo de memória.
- Disco: Há sobrecarga de espaço proporcional ao número de arquivos temporários gerados, que são removidos ao final da execução

Problemas e Observações:

Durante o desenvolvimento, alguns cuidados foram necessários:

- A ordenação correta exige que os tipos de dados nas colunas estejam coerentes (ex: números devem ser convertidos para float ou int).
- Para ordenações decrescentes, foi preciso adaptar as chaves no heap invertendo seu valor numérico ou lexicográfico.
- A leitura e escrita em arquivos CSV exigiram atenção à codificação (utf-8) e ao uso do módulo csv para evitar erros de formatação.

  Em alguns sistemas operacionais (como Windows), a remoção de arquivos temporários pode falhar se não forem fechados corretamente — por isso todos os arquivos são explicitamente fechados antes da exclusão.

Conclusão:

O algoritmo de ordenação externa desenvolvido demonstrou-se funcional, eficiente e adequado ao problema proposto. A estratégia de dividir o arquivo original em runs que cabem na memória e intercalá-los com um heap permitiu escalar a ordenação para arquivos de qualquer tamanho, respeitando os limites da memória RAM. A modularização clara do código facilita futuras melhorias, como paralelização dos merges, compressão dos arquivos temporários ou uso de bibliotecas como pandas para leitura seletiva. O conhecimento de algoritmos clássicos como Merge Sort, aliado à manipulação de arquivos e estruturas como heaps, foi fundamental para a construção de uma solução sólida e escalável, com aplicações diretas em Big Data, Engenharia de Produção e Sistemas de Informação.