UNIVERSIDADE FEDERAL DE LAVRAS
INSTITUTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
SISTEMAS DE INFORMAÇÃO

(Grupo 24)
Cesar Augusto Pires
Gustavo Ribeiro de Figueiredo

Ordenação em Memória Secundária
Relatório do projeto prático

LAVRAS
2023

1. Sumário

2. Introdução e Objetivo

Primeiramente, gostaríamos de esclarecer a retirada do nome do aluno Caio do grupo, durante o percorrer do semestre por motivos pessoais o aluno acabou desistindo da disciplina de Estrutura de Dados, dessa forma como o mesmo não contribuiu com a segunda parte do trabalho em comum acordo decidimos remover o nome dele visto todo o cenário.

Este projeto tem como objetivo ordenar o arquivo "Subnational-period-life-tables-2017-2019-CSV" em ordem crescente, utilizando como atributos primários a área e como secundário os valores contidos no arquivo. O grupo 24 foi selecionado para realizar esta tarefa, e para implementar a solução foi escolhido o algoritmo de ordenação merge sort, capaz de ordenar grandes quantidades de dados de forma eficiente.

Para implementar essa solução, o grupo desenvolveu um sistema em C++, que lê o arquivo CSV, realiza a ordenação utilizando o método de merge sort e grava o resultado em um novo arquivo. Com essa solução, o grupo 24 pretende fornecer uma ferramenta útil para ordenação de grandes arquivos CSV em ordem crescente, permitindo uma melhor análise e manipulação dos dados contidos nesses arquivos.

Além disso, o objetivo dessa documentação é esclarecer os processos e decisões tomadas pelo grupo ao desenvolver o projeto.

3. Desenvolvimento

3.1. Struct (subnationalPeriodLifeTablesStruct.h)

Visando deixar o código mais reaproveitável criamos a struct dentro de um arquivo separado dessa forma para utiliza-la apenas precisamos importar o arquivo sem ter que a cada arquivo que ela é usada reescrever toda a declaração.

Esse arquivo define uma estrutura de dados chamada SubnationalPeriodLifeTables, que representa uma tabela de expectativa de vida em diferentes áreas geográficas. A estrutura contém 9 campos:

- **id** - um número inteiro que identifica de forma única cada linha da tabela.
- **measure** - uma string que descreve a medida da expectativa de vida (por exemplo, esperança de vida ao nascer ou expectativa de vida aos 65 anos).
- **quantile** - um número em ponto flutuante que representa a porcentagem da população com a expectativa de vida mais baixa (por exemplo, 10%).
- **area** - uma string que descreve a área geográfica para a qual a expectativa de vida foi calculada (por exemplo, um país ou estado).
- **sex** - uma string que descreve o sexo da população (por exemplo, masculino ou feminino).
- **age** - uma string que descreve a faixa etária para a qual a expectativa de vida foi calculada (por exemplo, recém-nascidos ou idosos).
- **geography** - uma string que descreve a região geográfica em que a área está localizada (por exemplo, América Latina ou Ásia).
- **ethnic** - uma string que descreve o grupo étnico ao qual a população pertence (por exemplo, branco ou afrodescendente).
- **value** - um número em ponto flutuante que representa a expectativa de vida em anos para a população descrita pela linha da tabela.

3.2. Conversão de CSV para Binário (conversor.h)

Primeiramente no nosso arquivo *conversor.h* temos a importação do arquivo *subnationalPeriodLifeTablesStruct.h.*

O arquivo de conversão é responsável por tratar a iteração do arquivo CSV para arquivo binário, nesse arquivo temos o método *converterCSV* que será o método principal do arquivo que terá a responsabilidade de iterar o arquivo CSV e também teremos uma classe chamada de *LinhaCsv* que téra como responsabilidade tratar a Linha para alimentar a struct que será salva. Nas sessões abaixo falaremos das funcionalidades dos métodos da classe e do método void.

3.2.1. Classe LinhaCsv

A classe LinhaCsv tem três membros públicos: atributoLinha, restanteLinha e dois métodos, tratarLinha e tratarPorcentagem.

3.2.1.1. Método tratarLinha

O método tratarLinha é responsável por receber uma string que contém uma linha do arquivo CSV e separar o primeiro campo até a primeira vírgula em atributoLinha, e o restante da linha em restanteLinha. Essa separação é realizada utilizando a função find para encontrar a posição da primeira vírgula e a função substr para pegar a substring desde o início até a posição da vírgula.

3.2.1.2. Método tratarPorcentagem

O método tratarPorcentagem recebe como entrada uma string, que pode conter um número seguido do símbolo de porcentagem (%), ou apenas um número. O método verifica se a última posição da string é o símbolo de porcentagem e, em caso positivo, remove o símbolo e divide o número por 100. Em seguida, converte a string para um número em ponto flutuante utilizando a função stof e retorna o resultado.

3.2.2. Método Void converterCSV

A função converterCSV() lê o arquivo CSV "SubnationalPeriodLifeTables.csv" e escreve os dados em um arquivo binário "SubnationalPeriodLifeTables.bin". A leitura do arquivo é feita linha por linha, utilizando a classe LinhaCsv para tratar as informações de cada linha. As informações são armazenadas em um objeto SubnationalPeriodLifeTables, que é escrito no arquivo binário utilizando o método write().

3.3. Auxiliar de Ordenação (auxiliarOrdenador.h)

Este arquivo contém um conjunto de funções que realizam operações em arquivos binários que armazenam dados de tabelas de vida subnacionais (SubnationalPeriodLifeTables). Abaixo estão as descrições de cada uma das funções:

3.3.1. Método obterTamanhoArquivo

Recebe como parâmetro um objeto ifstream que representa um arquivo binário de entrada e retorna o tamanho (em bytes) do arquivo.

3.3.2. Método obterQuantidadeRegistrosArquivo

Recebe como parâmetro o tamanho do arquivo (em bytes) e um objeto SubnationalPeriodLifeTables que representa um registro da tabela de vida subnacional. Retorna a quantidade de registros que o arquivo possui.

3.3.3. Método exibirInformacoesArquivo

Recebe como parâmetro o tamanho do arquivo e a quantidade de registros. Imprime na saída padrão uma mensagem informando o tamanho e a quantidade de registros do arquivo.

3.3.4. Método verificarFalhaAberturaArquivo

Recebem como parâmetro um objeto ifstream ou ofstream, respectivamente, que representam um arquivo binário de entrada ou saída. Verificam se houve falha na abertura do arquivo e retornam true em caso positivo ou false em caso negativo.

3.3.5. Método verificarFalhaAberturaArquivos

Recebe como parâmetro objetos ifstream e ofstream que representam arquivos binários de entrada e saída. Verifica se houve falha na abertura dos arquivos e em caso positivo, exibe uma mensagem de erro e encerra o programa.

3.3.6. Método tratarBlocosArquivo

Recebe como parâmetro o início e o fim de um bloco de registros no arquivo de entrada, um objeto SubnationalPeriodLifeTables que representa um registro da tabela de vida subnacional, e objetos ifstream e ofstream que representam arquivos binários de entrada e saída, respectivamente. A função copia os registros do bloco do arquivo de entrada para o arquivo de saída.

3.3.7. Método fecharArquivo

Recebem como parâmetro um objeto ofstream ou ifstream, respectivamente, e encerram o arquivo.

3.3.8. Método abrirAuxiliarEntrada

Recebe como parâmetros um objeto ifstream, um valor booleano ida e um vetor ultimo de booleanos. Dependendo do valor de ida e do conteúdo do vetor ultimo, a função abre um arquivo auxiliar de entrada ("arquivoTemporarioB1.dat", "arquivoTemporarioB2.dat",

"arquivoTemporarioC1.dat" ou "arquivoTemporarioC2.dat") para ser utilizado em uma operação posterior.

3.4. Ordenação Externa (ordenador.h)

O objetivo da ordenação externa é ordenar grandes quantidades de dados que não podem ser alocados na memória principal. Dessa forma, a ordenação é feita em blocos menores que cabem na memória e depois esses blocos são combinados em uma nova ordenação.

3.4.1. Método comparaChaves

A primeira função é a comparaChaves, que recebe dois registros SubnationalPeriodLifeTables e uma variável booleana que indica se a comparação é pela chave primária (true) ou pela chave secundária (false). A função compara os registros a partir da chave especificada e retorna um número negativo, zero ou positivo, indicando se o primeiro registro é menor, igual ou maior que o segundo registro.

3.4.2. Método intercalaBloco

A segunda função é a intercalaBloco, que recebe dois arquivos para leitura (auxEntrada[2]), dois arquivos para escrita (auxSaida[2]), um inteiro passo que indica o tamanho do bloco e um inteiro posicaoSaida que indica a posição do arquivo de saída para a escrita. A função realiza a intercalação de blocos ordenados, lendo os blocos de entrada e escrevendo no arquivo de saída.

3.4.3. Método mergeSortExterno

A terceira função é a mergeSortExterno, que é a função principal da ordenação externa. Ela realiza a leitura do arquivo de entrada, divide o arquivo em blocos menores e ordena esses blocos usando o algoritmo Merge Sort. Em seguida, realiza a intercalação dos blocos ordenados até que todo o arquivo esteja ordenado. O arquivo de entrada é o arquivo SubnationalPeriodLifeTables.bin, e o arquivo de saída é o arquivo SubnationalPeriodLifeTablesOrdenado.bin. Além disso, a função também cria dois arquivos temporários para armazenar os blocos intermediários da ordenação.

3.5. Auxiliar de Leitura (leitura.h)

3.5.1. Função print

A função print simplesmente imprime cada um dos campos do registro com um separador '|' entre eles.

3.5.2. Função busca

A função recebe como parâmetro o nome do arquivo binário que deve ser aberto e percorrido. Primeiro, a função abre o arquivo e verifica se foi possível abri-lo. Em seguida, ela determina o número de registros no arquivo, dividindo o tamanho do arquivo pelo tamanho de um registro. Depois, a função percorre o arquivo, registro por registro, lendo cada um e chamando a função print para imprimir os dados na tela.

3.6. Arquivo Principal (main.cpp)

Esse arquivo contém um programa principal que realiza a conversão de um arquivo CSV em binário, ordena os dados no arquivo binário e permite que o usuário visualize os dados desordenados ou ordenados.

O programa inclui três bibliotecas (conversor.h, ordenador.h e leitura.h) e usa a biblioteca iostream para entrada e saída de dados. Em seguida, o programa define duas constantes para o nome do arquivo desordenado e ordenado.

3.6.1. Função imprimirDados

A função imprimirDados é usada para perguntar ao usuário se ele deseja visualizar os dados desordenados ou ordenados, dependendo da variável booleana "ordenados" que é passada como argumento para a função. Essa função, por sua vez, chama a função "busca" que lê o arquivo binário e imprime seus dados na tela.

3.6.2. Função main

Na função main, primeiro é exibida uma mensagem informando que a conversão do arquivo CSV em binário está sendo realizada. Em seguida, o programa chama a função converterCSV para realizar a conversão. Depois, a variável booleana "ordenados" é definida como false e a função imprimirDados é chamada para exibir os dados desordenados.

Então, é exibida outra mensagem informando que a ordenação do arquivo está sendo realizada e a função mergeSortExterno é chamada para ordenar o arquivo binário. Em seguida, a variável booleana ordenados é definida como true e a função imprimirDados é

chamada novamente para exibir os dados ordenados. Por fim, a função "main" retorna 0.

4. Execução do Programa

Para executar o programa basta certificar que tenha o arquivo *SubnationalPeriodLifeTables.csv* no mesmo diretório do arquivo main e dos outros arquivos .h.

4.1. Linux

Para executar o programa no linux, após a compilação do programa basta executar no terminal o seguinte comando:

```
Unset
./main
```

4.2. Windows

Já no windows, basta executar o .exe gerado após a compilação

5. Conclusão

Em resumo, o objetivo principal do projeto foi alcançado, que era ordenar um arquivo CSV de grande tamanho de forma eficiente utilizando o algoritmo de ordenação merge sort. O sistema desenvolvido pelo grupo 24 em C++ é capaz de realizar a leitura, ordenação e gravação dos dados em um novo arquivo, utilizando como atributos primários a área e como secundário os valores contidos no arquivo.

Em suma, o sistema desenvolvido pelo grupo fornece uma ferramenta útil para ordenação de grandes arquivos CSV em ordem crescente, permitindo uma melhor análise e manipulação dos dados contidos nesses arquivos. Esperamos que essa documentação possa esclarecer os processos e decisões tomadas pelo grupo ao desenvolver o projeto e que possa ser útil para aqueles que desejam implementar soluções semelhantes.