Analise de um de dados do governo brasileiro referente ao vírus SARS-CoV-2

Reberth Kelvin Santos de Siqueira
Algoritmo e Estrutura de dados II – Turma: Integral
e-mail: reberthkss@outlook.com

Resumo. *SARS-Cov-2, mais conhecido como Corona Virus é um virus de origem asiática identificada no fim de 2019 pelas entidades chinesas na China. Este projeto foi desenvolvido como um trabalho a ser desenvolvido na materia de Algoritmo e Estrutura de Dados II visando avaliar os conceitos desenvolvidos nas aulas sobre métodos de ordenação com o objetivo de estabelecer métodos para analise dos dados contidos em um arquivo do tipo .csv.*

Introdução

Este projeto foi desenvolvido como um trabalho a ser desenvolvido na materia de Algoritmo e Estrutura de Dados II ofertada pela Universidade Federal de São Paulo – UNIFESP ministrada pela docente Lilian Berton.

A base de dados foi retirada do site de relatórios do governo brasileiro (disponível em: https://covid.saude.gov.br/) e importado para um arquivo no formato .csv para utilização em um programa escrito na linguagem de programação C.

O programa desenvolvido teve como objetivo obter os principais dados referente a pandemia do novo corona virus. Foram coletados dados diários de cada estado brasileiro, onde visou analisar os dados referente à número total de casos, número de novos casos por dia, número total de recuperados, número de novos recuperados por dia, número total de mortes, número total de mortes por dia, e número total de pessoas se recuperando.

Foi gerada uma opção de criação de um relatório que é salvo no formato de arquivo .txt para utilização em outras ferramentas de análise, como por exemplo o Excel, 3 opções de geração de relatório foram desenvolvidas, uma utilizando o algoritmo de ordenação quick sort, bubble sort e radix sort.

Procedimento Experimental

Para este projeto foi utilizado a linguagem C como ferramenta de programação e a IDE CLion da empresa Jetbrains.

Para definir como o programa seria feito, foi analisado a estrutura do arquivo com o objetivo de identificar padrões que possibilitassem criar um programa para salvar os dados. Após a análise realizada foi identificado que o arquivo tinha o caractere ‘;’ como identificador de fim de um campo do dataset; cada linha do dataset se referia a uma data e um estado; o dataset tem o campo cod_uf, este campo foi utilizado como ID único do dado manipulado, o campo data também será utilizado para armazenar os detalhes dos dados do progresso diário da doença.

A estrutura de dado

A estrutura desenvolvida visa coletar os dados de cada estado e armazenar os dados diarios (por data) de cada um. Assim, conforme a figura 1, foram desenvolvidos três estruturas: CovidSummaryData, CovidDetailedData e Date.

Figura 1 – Estruturas CovidSummaryData, CovidDetailedData e Date

Destaca-se a implementação de um ponteiro para a estrutura CovidDetailedData, este ponteiro tem como objetivo tornar possivel a criação de um vetor através da alocação dinamica. Cada item deste vetor será correspondente a uma data de coleta de dados de um determinado estado.

A aplicação da estrutura CovidSummaryData também foi feita utilizando um ponteiro com o

intuito de criar um vetor, onde cada item desse vetor é referente a um estado, conforme figura 2.

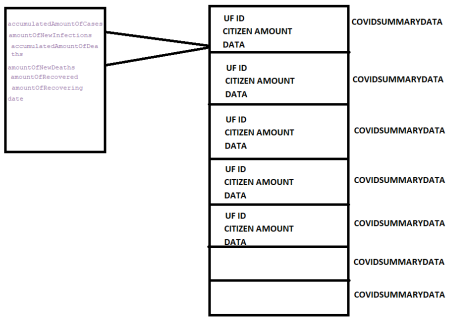

Figura 2 - Esquema da estrutura de dados

A lógica do programa

Como o identificador de termino de cada campo é o ponto e virgula, o programa foi desenvolvido seguindo essa definição. O programa lia cada caracter do documento, assim ele checa se o caracter lido é o ponto e virgula, se for ele realiza todo um procedimento para manipular os valore lidos previamente, se não o programa salva o caracter lido em um vetor de caracteres nomeado value.

Esse procedimento se repete até que o programa chegue no final do arquivo.

Abaixo a figura 3 e 4 que representam o fluxo do programa.

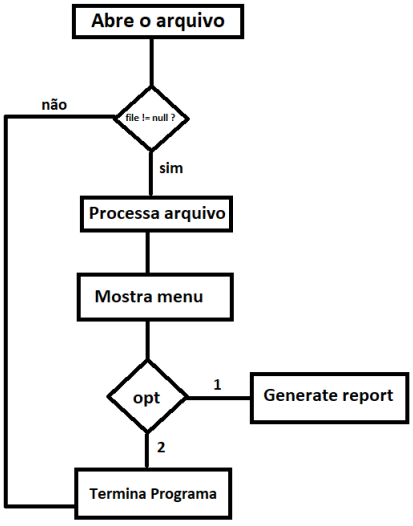

Figura 3 – Fluxo principal do programa

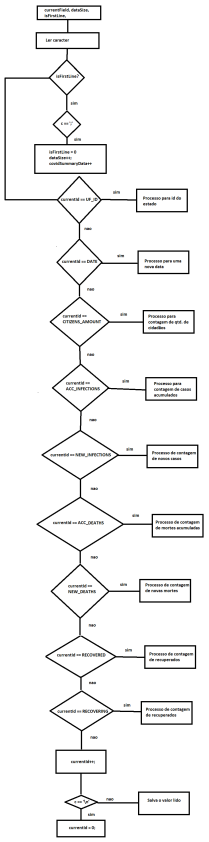

Figura 4 – Fluxo detalhado da função que processa o arquivo

Conforme ilustrado, o programa inicia processando o dataset a ser analisado. Dentro dessa função de processamento temos todo o processo descrito na Figura 4. O arquivo é aberto, um caractere do arquivo é obtido, o caractere é processado por uma checagem, se ele for qualquer caractere alfanumérico menos ‘;’ o programa salva o valor do caractere em um vetor de caracteres. Se o caractere for ‘;’ então significa que o programa terminou de ler um campo e deverá a partir dai processar o valor que está armazenado no vetor de caractere mencionado anteriormente. Outra verificação que é feita é a comparação com ‘\n’, se o caractere for ‘\n’ então significa que o programa chegou no final da linha e deverá realizar a limpeza das variáveis utilizadas, ou seja, “zerar” elas para a leitura da próxima linha.

Para cada leitura de campo o programa irá verificar a qual identificador aquele campo corresponde, os identificadores são:

UF_ID (col. 4), DATE (col. 8), CITIZENS_AMOUNT (col. 10), ACC_INFECTIONS (col. 11), NEW_INFECTIONS (col. 12), ACC_DEATHS

(col. 13), NEW_DEATHS (col. 15), RECOVERED (col. 16), RECOVERING (col. 17).

A seguir a descrição do que acontece quando o programa ser igual a um dos identificadores descritos acima:

UF_ID: Ao corresponder a esse campo, o valor contido dentro da variavel value é referente ao campo cod_uf, descrito dentro do arquivo .csv. Assim o programa verifica se já existe alguma estrutura CovidSummaryData alocada para o código do estado contido no vetor de caractere. Se já houver, o programa apenas salva o indice em que essa estrutura esta posicionada no vetor.

DATE: O valor contido no vetor de caractere quando o identificador é o date é o valor da data no formata DD/MM/YYYY, esse valor é convertido para uma estrutura date e logo em seguida é feita uma comparação para identificar se o dia correspondente na vetor de caracteres já existe e está alocado. Se não houve o programa aloca uma nova estrutura CovidDetailedData para a variavel data, senão o programa salva o index da posição em que a estrutura de dados representando aquele dia está armazenada.

Após passar pelos identificadores UF_ID e DATE, o programa já tem o index do estado e do dia que aquele dado contido no vetor de caractere corresponde (variaveis indexState e indexDate). A partir disso ele realiza uma soma para ir contabilizando novos carros ao decorrer do dia. Esse processo vale para aos identificadores CITIZENS_AMOUNT, ACC_INFECTIONS, NEW_INFECTIONS, ACC_DEATHS, NEW_DEATHS, RECOVERED, RECOVERING. Este processo se repete até que o arquivo seja totalmente processado.

O relatorio

Após processar o arquivo, o programa fornece um menu para gerar um relatório sobre os estados com **maiores** mortes no pais e outra opção para sair do programa.

Resultados

Testes de desempenho foram desenvolvidos medir o tempo de execução para a ordenação da mesma estrutura de dados, buscando ordenar dos estados com menor até maiores mortes. Foi realizado um teste com o algoritmo de ordenação quick, bubble e radix sort. Os resultados de tempo estão listados na tabela 1.

Alogoritmo de ordenação	Comple-xidade	Tempo (s)
Radix	O(n + k)	0.000000
Bubble	O(nlogn)	0.001000
Quick	$O(n^2)$	0.000000

Tabela 1 – Tabela de resultados da ordenação por maior número de mortos x estado

Para uma melhor analise do relatório gerado os dados contidos no arquivo Report_of_states_with_most_deaths.txt (gerado pelo programa) foram copiados e utilizados em um arquivo excel para plotagem de um gráfico de mortes x estado (gráfico 1).

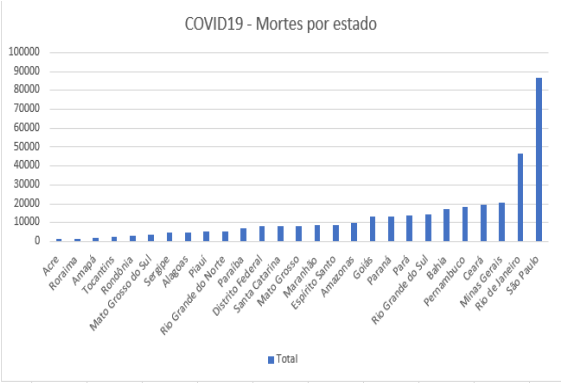

Gráfico 1 – Mortes x estado

Discussão

Com esta pesquisa foi possível identificar e estabelecer um padrão para análise de dados dos relatórios de alertas do covid 19 fornecidas pelo governo. A partir do programa é possível gerar diversos relatórios, o relatório escolhido foi o de mortes x estados, mas é possível gerar muitos outros.

Conclusão

Portanto o objetivo de gerar um programa para analisar os dados do arquivo .csv foram atingidos. Novos relatórios podem ser feitos e salvos com base no método e criterio de ordenação estabelecidos.

Referências:

S. Ed. Define Array in C, Stackoverflow, 2012-03-08, Disponível em: https://stackoverflow.com/questions/9846920/define-array-in-c, Acesso em: 12 de dezembro de 2020.

Guelfi, Everton, Tabela de Código de UF do IBGE, Tecnospeed, 2018, Disponível em: https://atendimento.tecnospeed.com.br/hc/pt-br/articles/360021494734-Tabela-de-

C%C3%B3digo-de-UF-do-IBGE, Acesso em: 12 de dezembro de 2020.

Um Programador, Manipulação do malloc() e realloc(), Stackoverflow, 2018-08-14, Disponível em: https://pt.stackoverflow.com/questions/322228/manipula%C3%A7%C3%A3o-do-malloc-e-realloc, Acesso em: 14de dezembro de 2020.