TRABALHO PRÁTICO 00

CONVERSOR DE IMAGENS

ARTHUR LINHARES MADUREIRA

2021031599

UNIVERSIDADE FEDERAL DE MINAS GERAIS

BELO HORIZONTE – MG – BRASIL

arthurlm1234@gmail.com

1) INTRODUÇÃO

O problema consiste em criar um programa capaz de receber e armazenar uma imagem PPM em uma estrutura de dados dinâmica. Deve-se poder, ainda, converter tal imagem em um arquivo PGM e escrevê-lo em um arquivo de saída. Para isso, serão utilizadas matrizes dinamicamente alocadas.

2) MÉTODO

O programa foi desenvolvido na linguagem C++, compilada pelo compilador G++ da GNU Compiler Collection. O computador utilizado possui 8gb RAM e processador Intel Core i5-8265U 1.60GHz

A implementação do problema teve como base a criação de uma classe Imagem. Essa classe possui a seguinte modelagem:

- **ATRIBUTOS:**

- int altura
- int largura
- int cor
- string tipo
- RGB **ppm(sendo RGB uma struct com os atributos do tipo unsigned char r,g e b)
- unsigned char **pgm

- **MÉTODOS:**

- Imagem (int altura, int largura): método construtor.
- Ler (string nome_arquivo): lê o arquivo com o nome fornecido e o salva em uma matriz dinâmica.
- Converter (): realiza a conversão da imagem, que é armazenada na matriz pgm.
- Escrever(string nome_arquivo): cria um arquivo com o nome fornecido e escreve o conteúdo da matriz pgm nele.

3) ANÁLISE DE COMPLEXIDADE

Para realizar a análise de complexidade, vamos considerar que altura e largura têm o valor n.

3.1) TEMPO

- LER

Para esse método, devemos alocar a matriz dinâmica com um laço for que tem n iterações. Além disso, deve-se ler o arquivo PPM percorrendo todos os seus valores, o que equivale a n^2.

$$O(n) + O(n^2) = O(n^2)$$

- CONVERTER

Para converter, o arquivo deve-se percorrer todos os elementos da matriz dinâmica, ou seja, deve-se percorrer n^2.

$$O(n^2)$$

- ESCREVER

Para converter, o arquivo deve-se percorrer todos os elementos da matriz dinâmica, ou seja, deve-se percorrer n^2.

$$O(n^2)$$

3.2) ESPAÇO

Para calcular o espaço ocupado, deve-se levar em conta que o número de elementos da matriz é dado pela multiplicação da altura com a largura, isto é, $n*n = n^2$. Logo, o espaço ocupado é dado por $O(n^2)$.

4) INSTRUÇÕES DE COMPILAÇÃO E EXECUÇÃO

- Acesse o diretório TP0
- Utilizando um terminal utilize o comando “make”
- Então digite ./bin/run.out com as seguintes flags:
    - “-i” (obrigatória): coloque o nome do arquivo de entrada logo após essa flag.
    - “-o” (obrigatória): coloque o nome do arquivo de saída logo após essa flag.
    - “-p” (opcional): utilizada para iniciar o *memlog*, deve-se colocar, logo após a flag, o nome do arquivo onde serão armazenados os registros.
    - “-l” (opcional): caso essa flag seja usada, além do tempo de execução, também serão registrados os acessos à memoria.

6) CONCLUSÃO

Este trabalho lidou com a criação de um conversor de imagens, na qual foi utilizada uma matriz dinamicamente alocado para salvar as informações recebidas pelo arquivo PPM.

Com o procedimento utilizado, pode-se notar que é possível utilizar estrutura de

TRABALHO PRÁTICO 00

CONVERSOR DE IMAGENS

ARTHUR LINHARES MADUREIRA

2021031599

UNIVERSIDADE FEDERAL DE MINAS GERAIS

BELO HORIZONTE – MG – BRASIL

arthurlm1234@gmail.com

1) INTRODUÇÃO

O problema consiste em criar um programa capaz de receber e armazenar uma imagem PPM em uma estrutura de dados dinâmica. Deve-se poder, ainda, converter tal imagem em um arquivo PGM e escrevê-lo em um arquivo de saída. Para isso, serão utilizadas matrizes dinamicamente alocadas.

2) MÉTODO

O programa foi desenvolvido na linguagem C++, compilada pelo compilador G++ da GNU Compiler Collection. O computador utilizado possui 8gb RAM e processador Intel Core i5-8265U 1.60GHz

A implementação do problema teve como base a criação de uma classe Imagem. Essa classe possui a seguinte modelagem:

- **ATRIBUTOS:**

- int altura
- int largura
- int cor
- string tipo
- RGB **ppm(sendo RGB uma struct com os atributos do tipo unsigned char r,g e b)
- unsigned char **pgm

- **MÉTODOS:**

- Imagem (int altura, int largura): método construtor.
- Ler (string nome_arquivo): lê o arquivo com o nome fornecido e o salva em uma matriz dinâmica.
- Converter (): realiza a conversão da imagem, que é armazenada na matriz pgm.
- Escrever (string nome_arquivo): cria um arquivo com o nome fornecido e escreve o conteúdo da matriz pgm nele.

3) ANÁLISE DE COMPLEXIDADE

Para realizar a análise de complexidade, vamos considerar que altura e largura têm o valor n.

3.1) TEMPO

- LER

Para esse método, devemos alocar a matriz dinâmica com um laço for que tem n iterações. Além disso, deve-se ler o arquivo PPM percorrendo todos os seus valores, o que equivale a n^2.

$$O(n) + O(n^2) = O(n^2)$$

- CONVERTER

Para converter, o arquivo deve-se percorrer todos os elementos da matriz dinâmica, ou seja, deve-se percorrer n^2.

$$O(n^2)$$

- ESCREVER

Para converter, o arquivo deve-se percorrer todos os elementos da matriz dinâmica, ou seja, deve-se percorrer n^2.

$$O(n^2)$$

3.2) ESPAÇO

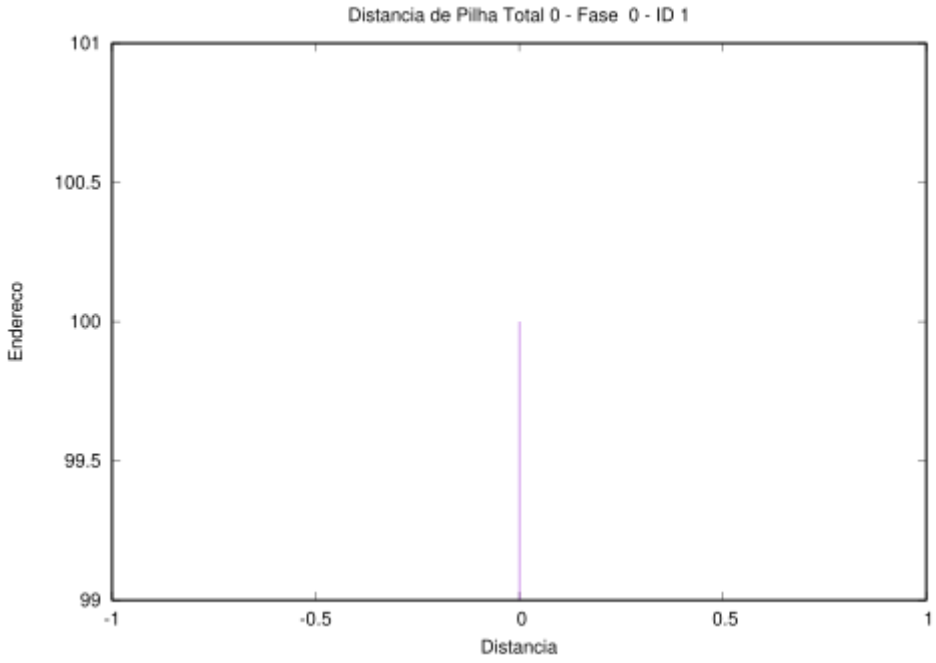
Para calcular o espaço ocupado, deve-se levar em conta que o número de elementos da matriz é dado pela multiplicação da altura com a largura, isto é, $n*n = n^2$. Logo, o espaço ocupado é dado por $O(n^2)$.

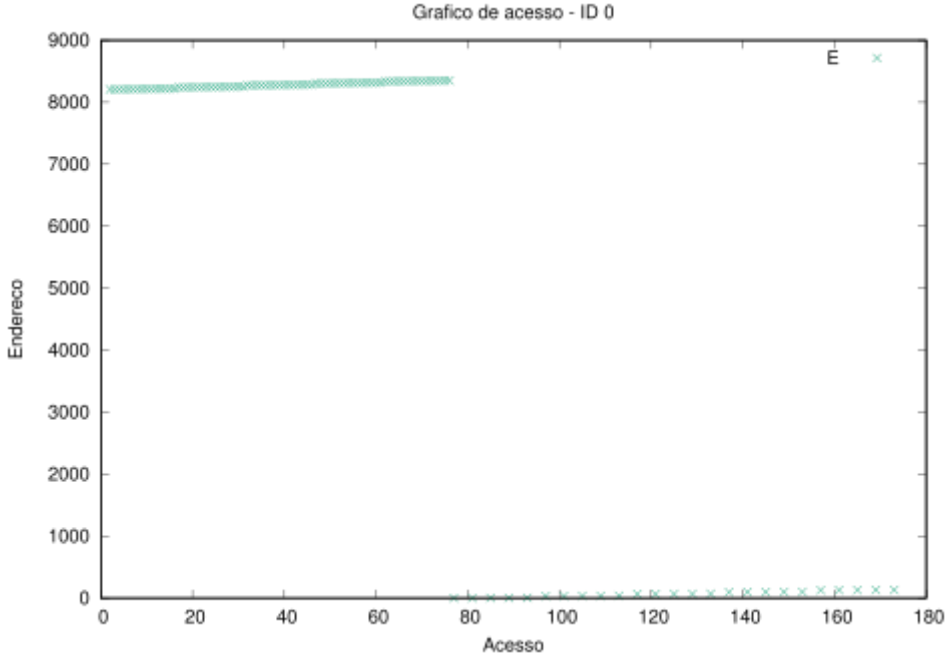
4) ESTRATÉGIAS DE ROBUSTEZ

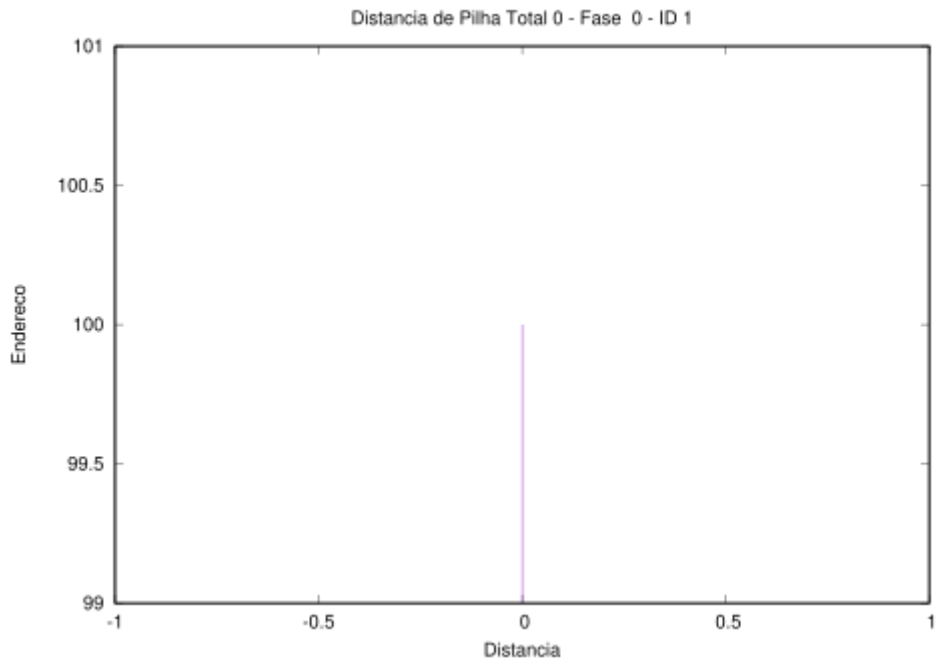
- Exigir flag “-i” para saber o arquivo de entrada;
- Exigir flag “-o” para saber o arquivo de saída;
- Não permite usar flag “-l” sem iniciar o memlog;
- Exige que o arquivo de entrada seja do tipo “P3”;
- Exige que a altura e a largura sejam positivas;

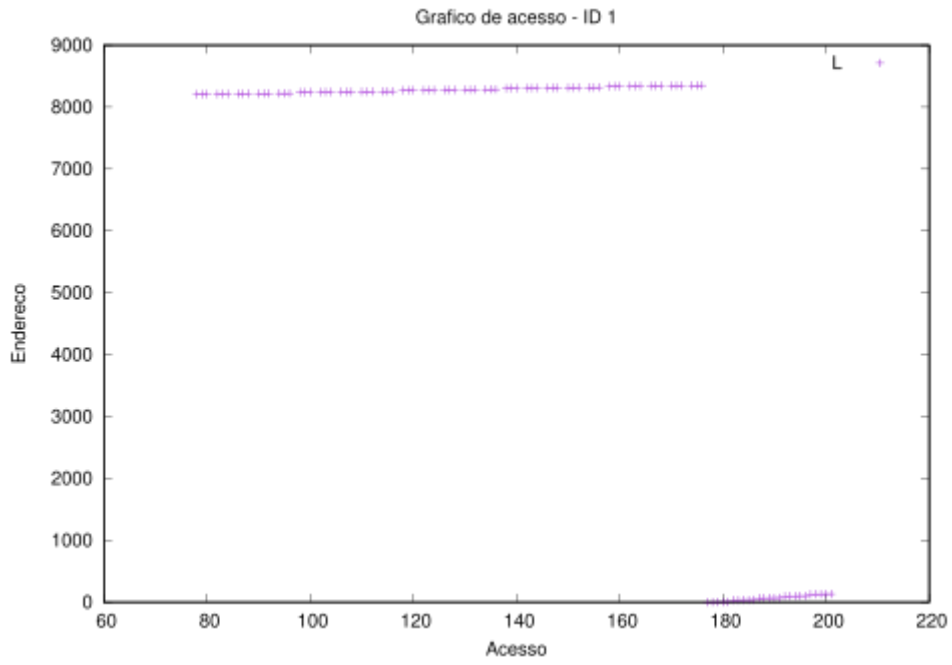
- Exige que o atributo “cor” do arquivo de entrada seja 255.

5) ANÁLISE EXPERIMENTAL









Cada vez que se acessa um endereço, o mesmo é colocado no topo da pilha. Por isso, a distância de pilha é 0.

Os gráficos de acesso mostram os dados sendo lidos e escritos nas matrizes pgm e ppm.

6) CONCLUSÃO

Este trabalho lidou com a criação de um conversor de imagens, na qual foi utilizada uma matriz dinamicamente alocação para salvar as informações recebidas pelo arquivo PPM.

Com o procedimento utilizado, pode-se notar que é possível utilizar estrutura de dados dinâmicas para armazenar informações de arquivo de texto e, dessa forma, manipulá-los.

7) INSTRUÇÕES DE COMPILAÇÃO E EXECUÇÃO

- Acesse o diretório TP0
- Utilizando um terminal utilize o comando “make”
- Então digite ./bin/run.out com as seguintes flags:
    - “-i” (obrigatória): coloque o nome do arquivo de entrada logo após essa flag.
    - “-o” (obrigatória): coloque o nome do arquivo de saída logo após essa flag.
    - “-p” (opcional): utilizada para iniciar o *memlog*, deve-se colocar, logo após a flag, o nome do arquivo onde serão armazenados os registros.

“-l” (opcional): caso essa flag seja usada, além do tempo de execução, também serão registrados os acessos à memória