Universidade Federal do Espírito Santo

Departamento de Computação e Eletrônica - CEUNES
Estrutura de Dados I
Prof. Oberlan Romão

Exercício Programa 1

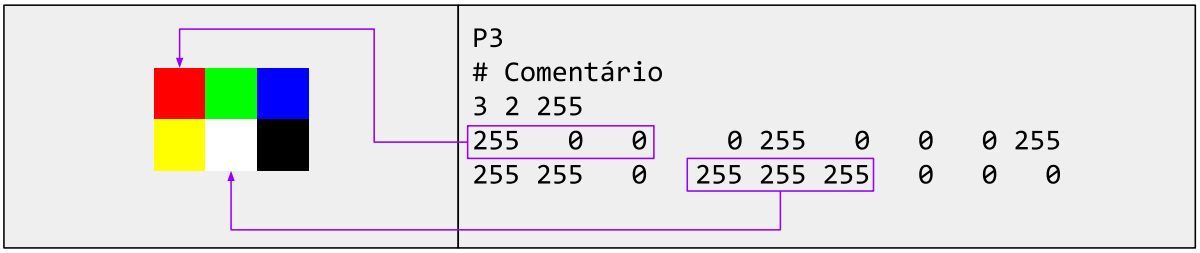
FotoXop

Uma imagem é basicamente uma matriz, com $altura$ (número de linhas) e $largura$ (número de colunas), e cada elemento da matriz é chamado de pixel (*picture element*), que possui uma "cor". Na sua forma mais básica podemos representar um pixel como aceso ("branco"), ou apagado ("preto"). Essas imagens são chamadas de binárias e podem ser representadas de forma bem compacta, já que precisamos apenas de 1 bit por pixel. No entanto, muitas vezes, dois níveis são insuficientes para representar uma imagem em "preto e branco", pois estas costumam possuir vários níveis ou tons de cinza. Uma alternativa é representar uma imagens em tons de cinza utilizando um byte (8 bits) para cada pixel. Assim, podemos representar imagens com até 256 níveis de cinza por pixel. Já uma imagem colorida requer mais informação para cada pixel. A representação mais comum é obtida decompondo uma cor nas componentes *red* (vermelho), *green* (verde) e *blue* (azul) ou RGB. A partir dessas cores podemos representar um grande espectro cromático de cores, por isso elas são chamadas de cores primárias.

Nesse exercício programa (EP) as imagens usadas estão codificadas no formato PPM (Portable pixmap), que é um formato bitmap, pixel a pixel, sem nenhum tipo de compressão. Imagens desse tipo podem ser lidas e gravadas por alguns editores de imagens, como o GIMP.

Imagens gravadas no formato PPM contêm os valores de cada pixel da imagem, linha por linha. Nesse formato, as cores dos pixels são gravadas como números de 0 a 255. A primeira linha contém o tipo da imagem. No EP, usaremos apenas o tipo `P3` — imagem colorida ASCII. A segunda linha (e todas as seguinte que começarem com "#") contém algum comentário e podem ser descartadas. A terceira linha contém três valores, que indicam a dimensão da imagem, ou seja, largura e altura em pixels, e o valor máximo do pixel (normalmente 255). Em seguida virão vários números, todos de 0 a 255, que indicam a cor de cada pixel. A quantidade depende do tamanho da imagem.

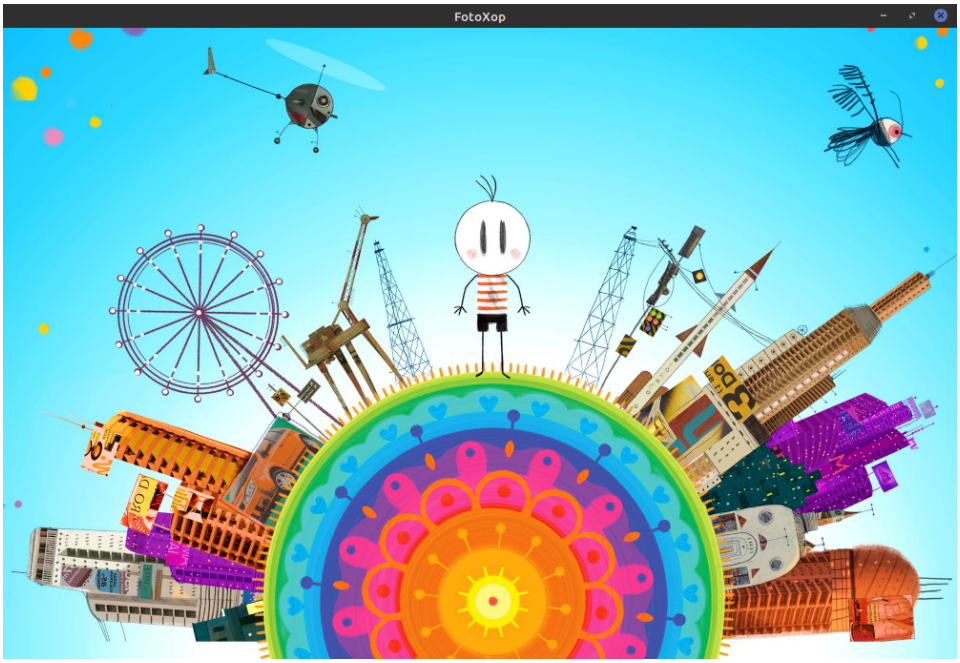
Considerando que a imagem é colorida, haverá pixels, cada um deles representado por três valores, todos entre 0 (min) e 255 (max). Esses 3 valores representam respectivamente a intensidade de Vermelho, Verde e Azul do pixel (padrão RGB). Veja o exemplo abaixo: 255 0 0 indica R=255, G=0, B=0, ou seja, vermelho puro; 0 0 0 indica preto, 255 255 255 branco, 255 255 0 indica amarelo, etc. Na figura, a parte da esquerda é o resultado visto ao abrir o arquivo em um visualizador de imagens. A esquerda é mostrado o conteúdo ao abrirmos a imagem em um editor de texto (VSCode, por exemplo).



Tarefa

Como você já deve imaginar, sua tarefa nesse EP é implementar um TAD que envolve algumas operações com imagens do tipo PPM. No AVA existe um "esqueleto" dos códigos, seu objetivo é completar as funções que ainda não foram implementadas. No EP usaremos OpenGL + GLUT para termos um programa mais iterativo.

O nome do executável produzido pelo `Makefile` disponibilizado é EP1. Para executar esse programa é necessário que seja fornecida na linha de comando o nome de um arquivo com uma imagem no formato PPM. Você pode usar o GIMP para gerar arquivos nesse formato. Fornecendo ao programa um arquivo com uma imagem uma janela é aberta com a visualização da imagem. Veja um exemplo abaixo:



Além de uma lista de opções, algumas mensagens são exibidas no terminal. Essas mensagens indicam, por exemplo, quais funções que ainda não foram implementadas. Ao completar uma função, remova o aviso.

O que entregar

Você deve entregar, pelo **AVA**, um arquivo compactado contendo todos os códigos. Você também deve enviar um relatório (arquivo no formato pdf) contendo alguns exemplos de execução do seu programa. Esse relatório deve conter uma explicação da função `meuFiltro`, detalhando qual filtro/efeito foi aplicado a imagem. Além disso, também deve conter exemplos dos resultados obtidos pelo seu programa, ou seja, imagens originais e o resultado obtido por cada função implementada.

Data de entrega: até às 6h do dia 27/03/2021.

Observações:

1. Não mude a estrutura dos arquivos enviados. Você deve entender e completar o código. Se precisar adicionar alguma função auxiliar, adicione no arquivo de implementação (arquivo `.c`);
2. Preferencialmente, use o Linux para a implementação. No Windows, você pode usar o Code::Blocks para auxiliar na compilação do projeto;

3. Seu código deve ser compilado com as flags:

   ```
   -O0 -std=c11 -Wall -Werror -Wextra -Wno-sign-compare -Wno-unused-parameter -Wno-unused-variable -Wshadow
   ```

   O `Makefile` e o projeto do Code::Blocks já foram configurados para usarem essas flags;
4. Acho que nem precisava dizer, mas EPs com erros de sintaxe (que não compilem) receberão nota 0;
5. Não mude o nome dos arquivos nem acrescente novos arquivos. A correção será feita usando o `Makefile` disponível no arquivo `FotoXop_Linux.zip`.
6. Código com vazamento de memória, valerá 70% da nota do EP;
7. Código que não segue o Guia de Estilo, valerá 90% da nota do EP;
8. O relatório deve ser entregue em formato pdf;
9. Use outras imagens além do arquivo `boy.ppm` disponibilizada;
10. Após aplicar um filtro e salvar a imagem, use o GIMP para exportá-la para o formato png ou jpeg;
11. **Em caso de plágio, será atribuído 0 a todos os envolvidos.**

Critérios de avaliação

A nota do EP se dará pela seguinte fórmula:

$$(1 - P) \times R \times (M \times G \times N_{EP} + N_R),$$

onde,

- $P = \begin{cases} 1, & \text{se houve plágio;} \\ 0, & \text{caso contrário.} \end{cases}$
- $R = \begin{cases} 1, & \text{se entregou o relatório como solicitado;} \\ 0, & \text{caso contrário (ou se enviou um arquivo em branco).} \end{cases}$
- $M = \begin{cases} 1.0, & \text{se o código não possui vazamento de memória;} \\ 0.7, & \text{caso contrário.} \end{cases}$
- $G = \begin{cases} 1.0, & \text{se seguiu o Guia de Estilo;} \\ 0.9, & \text{caso contrário.} \end{cases}$
- N_{EP}: Nota geral do EP, sendo $0.0 \leq N_{EP} \leq 8.0$;
- N_R: Nota geral do Relatório (caprichem!), sendo $0.0 \leq R \leq 2.0$.

Ponto extra para os melhores trabalhos :)

"Programming is not about typing, it's about thinking." - Rich Hickey