Exercício Programa 1 – Orientação a Objetos
Manipulação de Imagens PGM

Luan Guimarães Lacerda / Matrícula 12/125773
Professor: Paulo Meirelles

1. Objetivo

Construir um software na linguagem C++ que aplica três diferentes filtros em imagens do tipo PGM utilizando uma abordagem orientada a objetos e técnicas de controle de versão.

2. Solução

2.1. Fase Zero

Nessa etapa criou-se os diferentes diretórios onde seriam armazenados headers, binários, arquivos objetos, implementações dos headers e documentos referentes ao projeto.

Também foram criados os arquivos README.md e Makefile.

2.2. Fase Um

Com a organização estabelecida, iniciou-se a etapa de modelagem do projeto, analisando o problema e utilizando conceitos básicos de UML para definir as classes, seus atributos e métodos.

Adotando uma proposta de desenvolvimento na qual o software poderá um dia sofrer atualizações, podendo trabalhar com outros tipos de imagem como PBM e PPM, criou-se uma classe chamada **Imagem**, no qual estão definidos atributos como o número mágico, um ponteiro para os pixels e as dimensões da imagem sendo que este último é um objeto do tipo **Dimensoes**, classe que agrega a altura e a largura. A classe imagem também define alguns métodos abstratos como salvar e abrir arquivos.

Para trabalhar com imagens PGM definiu-se a classe **ImagemPGM** que herda de Imagem. Essa define alguns novos atributos referentes à esse tipo de arquivo como o nível máximo de cinza e comentários. Além disso, implementa os métodos abstratos da superclasse.

Implementou-se nesse ponto um Makefile mais genérico para poder compilar e testar as partes do programa sem ter que redefiní-lo todas as vezes.

2.3. Fase Dois

Com a classe ImagemPGM construida, restava implementar as classes de cada filtro. Para isso, definiu-se uma classe **Filtro** que possui métodos generalistas de aplicação de filtros em objetos do tipo ImagemPGM e alguns atributos como a mascara espacial, tamanho, divisor e limite.

Como as classes filhas **FiltroSmooth** e **FiltroSharpen** diferem apenas no conteúdo de seus atributos, os quais foram herdados de Filtro, bastou, no construtor de cada uma, implementar inicializações com valores definidos e diferentes.

Já a classe **FiltroNegativo**, apesar de possuir uma forma de aplicação completamente diferente, também herda de Filtro a fim de se utilizar os conceitos de polimorfismo, porém sobrescreve seus métodos de tratamento de imagem.

Todos os métodos de aplicação de filtros recebem como parâmetro ao menos um objeto do tipo ImagemPGM, o qual, ao final, terá apenas os valores de seus pixels modificados.

2.4. Fase Quatro

Com todas as classes funcionando como o previsto, implementou-se uma simples função principal **main** que instanciava cada classe e perguntava ao usuário que tipo de filtro ele desejava aplicar. Porém, posteriormente, decidiu-se contruir um menu mais sofisticado de opções, que disponibilizava aplicação de mais de um filtro na mesma imagem, salvar imagem modificada e testar outros filtros na imagem original, etc. Contruiu-se então uma classe **Menu** que agrega todos os métodos de interação com o usuário, fazendo validação de entradas e imprimindo saídas esperadas no terminal. A main ficou responsável apenas por instanciar e inicializar o menu.

3. Repositório no Gitlab

https://gitlab.com/luanguimararesla/ExercicioProposto1