Raster Image Processing

**Глава 1. Увод**

Проектът представлява конзолен редактор на растерни изображения. Редакторът поддържа работа с три вида файлови формата: *PPM*(*Portable PixMap*),   
*PGM*(*Portable GrayMap*) и *PBM*(*Portable BitMap*). При стартиране на приложението потребителят има възможността да създаде сесия/и, да прилага различни трансформации върху всички изображения от текущата сесия и, разбира се, записване на резултата от приложените трансформации. Идеята на проекта е да улесни и увеличи ефикасността при работата с изброените по-горе типове.

Целите ми относно този проект са да направя по-лесна работата с растерни изображения. Реализацията на основните операции като изрязване, завъртане и създаване на колаж от даден тип са ми главен преоритет, тъй като това може би са   
най-важните трансформации, които могат да бъдат извършени върху какъвто и да е вид изображение.

Документацията е структурирана по следния начин:

* **Глава 1. Увод**
* **Глава 2. Проблеми по време на разработване**
* **Глава 3. Проектиране**
* **Глава 4. Реализация и тестване**
* **Глава 5. Заключение**

**Глава 2. Проблеми по време на разработване**

При писането на проект е обичайно сблъскването с различни проблеми по време на разбротването му. Въпреки че всеки проект е уникален, някои от най-честите предизвикателства включват:

* **Грешки в кода:** Това включва синтактични грешки, логически грешки и т.н., които в повечето случаи са лесно поправими.
* **Грешно проектиране на архитектурата:** Неправилното проектиране води до проблеми с разширяемостта, поддръжката и четимостта на кода, но чрез спазването на добрите ООП практики лесно се избягва този проблем.
* **Интеграция и оптимизация на различни компоненти:** В някои случаи проектът може да има забавяне по време на изпълнение или да консумира прекалено много ресурси, както и да има преповтаряне на функционалността на вече съществуваща функция. В моя случай, при добавянето на различните видове растерни формати, осъзнавах колко неподходящо бяха проектирани част от функциите, извършващи трансформации на изображения.

Тези типове грешки лесно бяха отстранени чрез провеждането на систематични тестове.

Считам, че по специфичните проблеми, с които се сблъсках по време на писането на прoекта, са:

* **Загуба на йерархията спрямо видовете изображения:** Наложи ми се да избирам между темплейтен базов клас и шаблона *Factory.* При съвместното им използване се губеше йерархията, затова избрах употребата на *Factory,* защото прецених, че ще ми бъде по-полезно при разширяването на проекта в бъдеще.
* **Реализиране на *BMP*(*Bitmap format*):** Поради липсата ми на познания относно този формат, реших да не го добавям към проекта и да го оставя като бъдещ ъпдейт на програмата.
* **Големите *if*…*else*…блокове, служещи за създаване на нови инстанции на дадено растерно изображение:** Разреших този проблем чрез използването на шаблона *Factory,* който улеснява създаването на нова инстанция.
* **Големите *if*…*else*…блокове за изпълняване на команди:** За жалост не успях да избегна проверките на всяка команда, използвайки *if*…*else*….
* **Работа с едно и също изображение:** Решението на този проблем е да работя върху копие на оригинала на дадено изображение, след като е вкарано в програмата повторно.

**Глава 3. Проектиране**

Архитектура:

* **Image**(Abstract base class):  
  **Наследници:** pbm, pgm и ppm  
  **Характеристики:** name, extentionType, width и height
* **pbm**(Deriving class):  
  **Характеристики:** Като на *base*класа и pixels от тип *bool*
* **pgm**(Deriving class):  
  **Характеристики:** Като на *base*класа и pixels от тип *size\_t*
* **ppm**(Deriving class):  
  **Характеристики:** Като на *base*класа и pixels от тип *rgb*
* **rgb:  
  Характеристики:** red, green, blue и MAX\_VALUE
* **Session:  
  Характеристики: ID, images и imagesPaths**
* **Editor**(Singleton class):  
  **Характеристики:** sessions и loadedImages

Трите вида изображения (*pbm*, *pgm* и *ppm*) биват пазени в дадени сесии, които биват пазени в *Editor* класа. Чрез класа *Editor* биват изпълнявани команди, които правят промени върху изображенията.

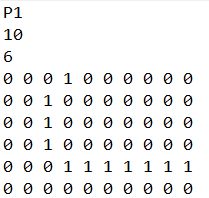
Относно разширяване на програмата, мисля че съм поставил добра основа, която би ми позволила лесно добавяне на нови видове изображения, без да променям логиката на кода и като цяло нещо по стария код, и също добавянето на нова функционалност, презиползваща вече съществуващи методи.

**Глава 4. Реализация и тестване**

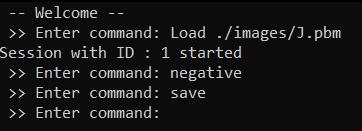
Може би най-важната част от реализирането на програмата е правилното четене и записване на растерните изображения, тъй като това е главната идея на този проект.

Пример 1:

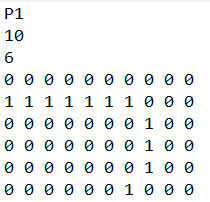
Image: J.pbm



След извършването на следните команди

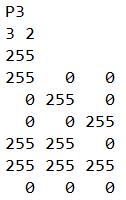


, изображенито ще изглежда така:

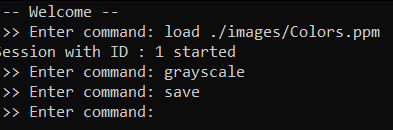


Пример 2:

Image: Colors.ppm

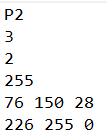


След извършването на следните команди



, изображенито ще изглежда така:

Image: Colors.pgm



**Глава 5. Заключение**

Мисля, че единствените неща, които за сега не съм направил, са работа с *BPM* файлове и работата с едно и също изображение. Най-вероятно тези две неща ще бъдат добавени в бъдещи ъпдейти на проекта.

**Github:** https://github.com/MiroslavVelikov/RasterImageProcessing