**Софийски университет „Св. Климент Охридски“**A black and white logo

Description automatically generated

*Факултет по математика и информатика*

*Дисциплина: “*Обектно-ориентирано програмиране*”*

**Тема: Обработка на растерни изображения**

**Изготвил:**

Александър Славов 4MI0700031

**Съдъражание**

[**1.Въведение в проектa** 3](#_Toc201341950)

[**2.Архитектура и дизайн** 3](#_Toc201341951)

[**2.1.Подход към решението** 3](#_Toc201341952)

[**2.2.Помощни структури за съхранение на метаданните на изображенията** 4](#_Toc201341953)

[**2.3. Полиморфна йерархия за съхранение и управление на изображения** 4](#_Toc201341954)

[**2.4. Полиморфна йерархия за управление на трансформациите прилагани върху изображенията** 6](#_Toc201341955)

[**2.5.Управление на сесии и потребителски интерфейс** 7](#_Toc201341956)

[**2.5.1.Управляване на сесии в приложението** 7](#_Toc201341957)

[**2.5.2.Потребителски интерфейс** 8](#_Toc201341958)

[**2.6.Клас диаграма на проекта** 9](#_Toc201341959)

[**3.Използвани ресурси** 9](#_Toc201341960)

[**4.Примери за използване на редактора** 11](#_Toc201341961)

[**4.1.Базов интерфейс на приложението** 11](#_Toc201341962)

[**4.2.Зареждане на сесия и прилагане на трансформации върху изображения** 11](#_Toc201341963)

# **1.Въведение в проектa**

Целта на настоящия проект е разработването на конзолно приложение за обработка на растерни изображения. Приложението функционира като текстов редактор на изображения, предоставяйки възможност на потребителите да зареждат, добавят, модифицират, съхраняват и управляват различни изображения в рамките на отделни сеси. Поддържаните фаилови формати са PPM, PGM, PBM.

Основните операции, които редактора предоставя са:

* Зареждане на изображения в сесия
* Трансофмиране на изображението в черно-бял вид
* Трансформиране на изображението в монохронен вид
* Трансофрмиране на изображението в негативен вид
* Завъртане на изображението 90 градуса наляво
* Завъртане на изображението 90 градуса надясно
* Завъртане на изображението огледално отгоре надолу
* Завъртане на изображението огледално отляво надясно
* Записване на обработените изображения с име по подразбиране
* Операция undo за премахване на последната операция
* Операция redo за премахване на последната undo операция
* Записавне на обработените изображения с име зададено от потребителя
* Информация за текущата сесия
* Изход от програмата.

Цялостна клас диаграма на проекта може да намерите в архива предаден заедно с решението и документацията на проекта!!!

# **2.Архитектура и дизайн**

## **2.1.Подход към решението**

Подхода, който сметнах за най-оптимален предвид задачата и изискванията определени върху нея, беше да разделя решението си на следните етапи:

* Създаване на помощните типове данни Matrix, Pixel, Resolution
* Създаване на полиморфна йерархия за съхранение и управление на изображения
* Създаване на полиморфна йерархия за извъшрване на трансформации
* Създаване на клас за управление на сесии
* Реализация на клас за създаване на изображения (ImageFactory)
* Разработване на конзолен интерфейс

В следващите точки подробно са описани причините и начина по който съответните етапи са реализирани.

## **2.2.Помощни структури за съхранение на метаданните на изображенията**

За съхранение метаданните, с които всяко изображения разполага бяха разработени следните типове данни:

* Pixel – Представлява пиксел с трицветна стойност в RGB формат, всеки компонент (red,green,blue) може да пази стойност между 0 и 65535 (unsigned short), този диапазон беше определен сългасно стандарта на изображенията от вид netpbm
* Resolution – Описва размерите на изображението, разполага с две член данни – width и height – съответено за широчината и височината на изображението
* Matrix – Шаблонен клас, който реализира матрица с достъп до елементите чрез оператор[][], в себе си съдържа вложения клас Proxy, който позволява синтаксиса [][] да бъде валиден

Клас диаграма на помощните струтури от данни изглежда по следния начин:

A screenshot of a computer

AI-generated content may be incorrect.

## **2.3. Полиморфна йерархия за съхранение и управление на изображения**

Представената отдолу йерархия представлява интерфейс за работа с изображения във форматите PBM, PGM, PPM които са част от NetPBM формата. Йерархията ни позволява да валидираме, зареждаме, запазваме и извличаме данни от изображенията във валиден вид, без да допускаме некоректен формат, непълнота или грешни метаданни да се зареждат. Йерархията се разделя на следните класове:

* Базов абстрактен клас NetPBMImage - дефинира общата функционалност за всички NetPBM формати. Той служи като основа за създаване, зареждане, модифициране и записване на изображения.
* Клас PBMImage - наследява NetPBMImage и представлява монохромно изображение (черно и бяло), където всеки пиксел е или 0, или 1.
* Клас PGMImage - представлява грейскейл изображение – сива скала, при която всеки пиксел има стойност от 0 (черно) до maxPixelValue (бяло).
* Клас PPMImage – Той моделира цветни изображения, където всеки пиксел съдържа стойности за червено, зелено и синьо, максималната стойност на всеки канал се определя чрез maxPixelValue
* Клас ImageFactory – Централизира логиката по създаване на подходящи изображения

Клас диаграма на полиморфната йерархия NetPBMImage:

A screenshot of a computer

AI-generated content may be incorrect.

## **2.4. Полиморфна йерархия за управление на трансформациите прилагани върху изображенията**

Йерархията предоставя абстракция за прилагане на различни модификации върху изображенията. Всяка трансформация е реализирана като наследник на абстрактния клас ImageTransformer, чрез прилагане на полиморфизъм, позволяващ еднакъв интерфейс за всички видове тансформации. Йерархията се разделя на следните класове:

* Абстрактен базов клас ImageTransformer – класа дефинира универсален интерфейс за трансформации чрез виртуалната функция apply(NetPBMImage&) и функцията clone, която създава динамично копие на трансформацията
* NegativeTransform - Създава негатив на изображението, обръщайки всеки цветен компонент на пиксела (R, G, B) спрямо максималната стойност.
* MonochromeTransform - Конвертира изображението в монохромно. Изчислява осреднена стойност (яркост) по формулата: 0.299 \* R + 0.587 \* G + 0.114 \* B
* GrayscaleTransform - Превръща изображението в сиво (грейскейл). Всеки пиксел получава еднаква стойност за R, G и B, изчислена по стандартната формула за яркост.
* FlipLeftTransform - Създава огледален образ по хоризонтала – разменя симетрични пиксели по колоните на всяка редица.
* FlipTopTransform - Създава огледален образ по вертикала – разменя симетрични редове на изображението.
* RotateLeftTransform - Завърта изображението на 90° наляво
  + Създава нова матрица с разменени размери (cols × rows);
  + Прехвърля стойностите от оригиналната в новата по нова логика на позициониране.
* RotateRightTransform - Завърта изображението на 90° надясно (по часовниковата стрелка) – огледално на предходната трансформация.

Клас диаграма на трансформациите:

A screenshot of a diagram

AI-generated content may be incorrect.

## **2.5.Управление на сесии и потребителски интерфейс**

### **2.5.1.Управляване на сесии в приложението**

Класът Session представлява логическа сесия за работа с еднo или повече изображения. Той поддържа списък с изображения, върху които се прилагат трансформации, както и механизъм за отмяна (undo) и възстановяване (redo) на действия. Всяка сесия има уникален идентификатор и поддържа операции като зареждане, добавяне на изображения, прилагане на трансформации и запис на резултата.

Клас диаграма на Session:

A screenshot of a computer program

AI-generated content may be incorrect.

### **2.5.2.Потребителски интерфейс**

Класът RasterImageEditor реализира команден интерфейс, чрез който потребителят взаимодейства с програмата. Той обработва входа от конзолата, разбира командите, валидира ги и ги насочва към съответните действия в текущата сесия. Този клас управлява състоянието на приложението и контролира цялостния процес на работа с изображения в рамките на една или повече сесии.

Клас диаграма на RasterImageEditor:

A screenshot of a computer program

AI-generated content may be incorrect.

## **2.6.Клас диаграма на проекта**

A screenshot of a computer

AI-generated content may be incorrect.

# **3.Използвани ресурси**

<https://en.cppreference.com/w/cpp/string/basic_string.html>

<https://en.cppreference.com/w/cpp/container/vector.html>

<https://en.cppreference.com/w/cpp/chrono.html>

<https://en.cppreference.com/w/c/chrono/time>

<https://docs.vultr.com/cpp/standard-library/ctime/strftime>

<https://refactoring.guru/design-patterns/factory-method>

<https://refactoring.guru/design-patterns/singleton>

<https://learn.fmi.uni-sofia.bg/course/view.php?id=10995>

# **4.Примери за използване на редактора**

## **4.1.Базов интерфейс на приложението**

A screenshot of a computer program

AI-generated content may be incorrect.

## **4.2.Зареждане на сесия и прилагане на трансформации върху изображения**

Направени са следните трансформации на четири подбрани изображения

A screenshot of a computer screen

AI-generated content may be incorrect.

Така излгежда едно от заредените изображения преди трансформация:

A screenshot of a computer

AI-generated content may be incorrect.

И така излгежда след прилагане на трансформациите:

A view from the bottom of a building

AI-generated content may be incorrect.

