Тема 11: Растерна графика

1. Увод

1.1 Описание

Реализираната програма позволява на потребителите да се създават сесии съдържащи снимки и да се прилагат операции върху съответните снимки. Има команден интерфейс, който позволява по-лесно задаване на желаните команди от потребителя. Програмата поддържа разширенията: .*pbm*, .*pgm* и .*ppm* в текстов формат.

1.2 Цел и задачи на разработката

Основни цели:

* Разработка на фунционален софтуер с подходяща архитектура
* Ефективно използване на ресурси като динамична памет
* Допълване на архитектурата с помощни класове и функции, които да осигурят по-добра четимост на кода и да улеснят имплементацията на основните задачи

Задачи на разработката:

* Създаване на потребителски интерфейс
* Възможност за създаване на нова сесия и работа с нея
* Обработка на снимките и записване на промените в подходящ тип файлове
* Откриване и имплементиране на подходящи валидации за съответните данни
* Имплементиране на общите функционалности

1.3 Структура на документацията

1. Увод…………………………………………………………………………………………..1

2. Преглед на предметната област…………………………………………………………….2

3. Проектиране………………………………………………………………………………….3

4. Заключение…………………………………………………………………………………...4

2. Преглед на предметната област

2.1 Дефиниране на проблеми и сложност на поставената задача

* Разбиране на условието – тъй като за пръв път се сблъсквам с този тип снимки, нямах идея как точно разботят те, което много затрудни разбирането на условието на заданието
* Проблеми с контрола на паметта – въпреки многото упражнения през целия курс, все още динамичната памет ме затруднява на моменти тъй като ме сблъсква с непознати за мен проблеми
* Проблеми с тестването – тестването на функции също беше проблем тъй като средата, която използвам създаваше проблеми и имаше няколко оттечки на памет, които бях пропуснала да поправя.

2.2 Подходи и методи решаване на поставените проблеми

* Проблемът с разбирането на условието беше решен чрез проучване какво точно представляват този тип снимки. За съжаление останаха няколко условия, които не успях да имплементирам, защото така и не успях да разбера как точно да ги включа в моята задача. Става въпрос за две от общите условия: load, което според мен върши същата работа като open (специфично за моята тема) и saveAs, което не разбирам какво точно трябва да прави тъй като програмата работи със сесии, които съдържат множество снимки и няма как да се запазят всички като една, нито има как да се запази сесията във файл, който да има логика.
* Справих се с проблема с контрола на памет като преразгледах щателно всички написани функции и успях да уточня къде точно се губят данни
* Проблемите с тестването бяха решени с влагане на много допълнително време

2.3 Потребителски и качествени изисквания

Ефикасна работа с динамичната памет, добре реализирана функционалност, спазване на принципите на ООП за абстракция и капсулация, интуитивна архитектура.

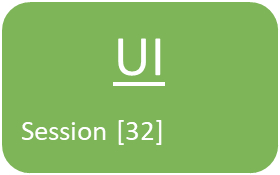
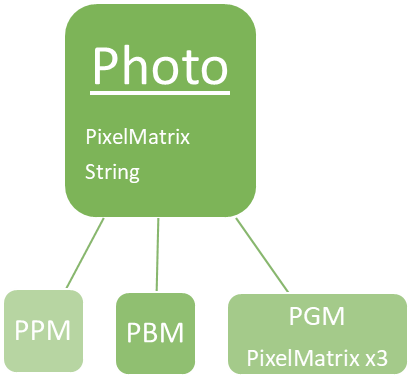
Глава 3. Проектиране

3.1 Обща архитектура – ООП дизайн

Програмата има 8 класа: *String, PixelMatrix, Photo, PBM, PGM, PPM, Session, UI*

Класът *String* представлява клас, описващ символен низ. Класът *PixelMatrix* обработва матрици от числа, които представляват пикселите на самите снимки. Класът *Photo* е абстрактен и се използва само за наследяване. Класовете *PBM, PGM, PPM* са наследници на класа Photo и те обработват данните за една снимка от съответния тип. Класът *Session* обработва сесиите и позволява на потребителя да използва функциите на класовете за снимки тъй като съдържа хетерогенен контейнер от снимки. Класът *UI* представлява конзолния интерфейс на програмата и позволява потребителя да редактира снимки от конзолата.

3.2 Диаграма, описваща програмата



Фигура 1

*Фигура1* е диаграма на програмата, която описва абстракцията ѝ. Двата класа *PixelMatrix* и *String* са помощни класове. В класа *Photo* се включва едн­а матрица от тип *PixelMatrix* и се използва *String* за задаване на име за съответната снимка. Класът *Photo* е абстрактен и има три наследника: *PBM, PGM, PPM.* Името на класовете описва типа снимка и се занимава с обработването на съответната снимка тъй като това се случва по различен начин за всеки тип. Класът *PPM* съдържа 3 матрици от тип *PixelMatrix* тъй като за всеки цвят работи с различна матрица. Класът *Session* съдържа хетерогенен контейнер от снимки и се занимава с извикването на правилните функции за съответните снимки. Класът *UI* се справя с конзолния интерфейс.

Глава 4. Заключение

4.1 Обобщение на изпълнението на началните цели

* Разработеният софтуер е с всички зададени функционалности и с добре разработена архитектура
* Динамичната памет е използвана ефективно и се освобождава на нужните места
* Кодът е четим тъй като бяха имплементирани нужните помощни класове.
* Липсват три от функционалностите:
  + *load –* функцията липсва, защото има същата функционалност като вече имплементираната функция *open.*
  + *saveAs –* функцията липсва, защото условието ѝ не е достатъчно определено за моята тема. Функцията трябва да запазва промените в даден файл. Тя обаче по логиката, която условието дава, трябва да се прилага върху сесия като останалите функции от общ тип. Това обаче няма как да стане, освен ако не се записват имената на снимките от сесията в споменатия файл, което няма смисъл. Може да се реализира с друга функционалност – да запазва само последната снимка.
  + *undo –* функцията липсва

4.2 Бъдещо развитие

Проектът може да се допълни с липсващите функционалности стига да се намери начин за логична имплементация. Също така в класа *PPM* не използва наследената от класа *Photo* матрица, това може да се коригира. Програмата поддържа само текстови файлове, но може да се направи да поддържа и бинарни. Могат да се добавят още функционалности като изрязване на част от снимката и промяна на гамата ѝ (не само в черно-бял стил). Също така програмата може да се препаботи така че да спазва *SOLID* принципите и изучените *Design-patterns.*

Линк за проекта в GitHub:

*https://github.com/GabrielaTopuzova/Raster-Graphics*