Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Программирование на языках высокого уровня

К ЗАЩИТЕ ДОПУСТИТЬ

Ассистент кафедры ЭВМ

А. В. Марзалюк

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

ГРАФИЧЕСКИЙ РЕДАКТОР ДЛЯ РАСТОВОЙ ГРАФИКИ

БГУИР КР 1-40 02 01 002 ПЗ

Студент Ю.Л.Спасёнов

Руководитель А.В.Марзалюк

МИНСК 2023

Учреждение образования

«Белорусский государственный университет информатики   
и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ЭВМ

Б. В. Никульшин

(подпись)

––––––––––––––– 20 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту    Спасёнову *Юрию Леонидовичу–––––––––––––––––––*

Тема проекта    *«Графический редактор для растовой графики»*

2. Срок сдачи студентом законченного проекта––––*11 декабря 2023 г.–*

3. Исходные данные к проекту   *Язык программирования: С++. файл”icons”.*

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке)

1. Лист задания.

2. Введение.

3. Постановка задачи

4. Обзор литературы.

4.1. Обзор методов и алгоритмов решения поставленной задачи.

5. Функциональное проектирование.

5.1. Структура входных и выходных данных.

5.2. Разработка диаграммы классов.

5.3. Описание классов.

6. Разработка программных модулей.

6.1. Разработка схем алгоритмов (два наиболее важных метода).

6.2. Разработка алгоритмов (описание алгоритмов по шагам, для двух методов).

7. Результаты работы.

8. Заключение

9. Литература

10. Приложения

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

*1. Диаграмма классов. ––––––––––––––––––––––––––––––*

*2. Схема алгоритма метода FilledShape.*

*3. Схема алгоритма метода gammaCorection.*

*––––––––––––––––––––––––––––––––––––––––––––– ––––––––––––––––*

6. Консультант по проекту (с обозначением разделов проекта)

А. В. Марзалюк

7. Дата выдачи задания –––––*15.09.2023г.––––––––––––––––––––––   –*

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапов):

*Выбор задания. Разработка содержания пояснительной записки. Перечень графического материала – к 01.10.2023 5 %;*

*разделы 2, 3 – к 01.10.2023 10 %;*

*раздел 4, 5 – к 01.11.2023 25 %;*

*раздел 6 – к 01.12.2023 35 %;*

*разделы 7,8,9 – к 05.12.2023 10 %;*

*раздел 10 – к 11.12.2023 5%;*

*оформление пояснительной записки и   
графического материала – к 11.12.2023 10 %;*

*Защита курсового проекта с 21.12 по 28.12.2023г.*

РУКОВОДИТЕЛЬ А.В. Марзалюк

(подпись)

Задание принял к исполнениюЮ.Л. Спасёнов

(дата и подпись студента)

Учреждение образования

«Белорусский государственный университет информатики   
и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ЭВМ

Б. В. Никульшин

(подпись)

––––––––––––––– 20 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту    Спасёнову *Юрию Леонидовичу–––––––––––––––––––*

Тема проекта    *«Графический редактор для растовой графики»*

2. Срок сдачи студентом законченного проекта––––*11 декабря 2023 г.–*

3. Исходные данные к проекту   *Язык программирования: С++. файл”icons”.*

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке)

1. Лист задания.

2. Введение.

3. Постановка задачи

4. Обзор литературы.

4.1. Обзор методов и алгоритмов решения поставленной задачи.

5. Функциональное проектирование.

5.1. Структура входных и выходных данных.

5.2. Разработка диаграммы классов.

5.3. Описание классов.

6. Разработка программных модулей.

6.1. Разработка схем алгоритмов (два наиболее важных метода).

6.2. Разработка алгоритмов (описание алгоритмов по шагам, для двух методов).

7. Результаты работы.

8. Заключение

9. Литература

10. Приложения

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

*1. Диаграмма классов. ––––––––––––––––––––––––––––––*

*2. Схема алгоритма метода FilledShape.*

*3. Схема алгоритма метода gammaCorection.*

*––––––––––––––––––––––––––––––––––––––––––––– ––––––––––––––––*

6. Консультант по проекту (с обозначением разделов проекта)

А. В. Марзалюк

7. Дата выдачи задания –––––*15.09.2023г.––––––––––––––––––––––   –*

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапов):

*Выбор задания. Разработка содержания пояснительной записки. Перечень графического материала – к 01.10.2023 5 %;*

*разделы 2, 3 – к 01.10.2023 10 %;*

*раздел 4, 5 – к 01.11.2023 25 %;*

*раздел 6 – к 01.12.2023 35 %;*

*разделы 7,8,9 – к 05.12.2023 10 %;*

*раздел 10 – к 11.12.2023 5%;*

*оформление пояснительной записки и   
графического материала – к 11.12.2023 10 %;*

*Защита курсового проекта с 21.12 по 28.12.2023г.*

РУКОВОДИТЕЛЬ А.В. Марзалюк

(подпись)

Задание принял к исполнениюЮ.Л. Спасёнов

(дата и подпись студента)

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 5](#_Toc20348)

[1. ПОСТАНОВКА ЗАДАЧ 6](#_Toc30656)

[1.1 Основные термины 6](#_Toc29739)

[2 ОБЗОР ЛИТЕРАТУРЫ 8](#_Toc19831)

[2.1 Обзор аналогов 8](#_Toc2905)

[2.2 Функциональные требования 9](#_Toc5142)

[2.3 Основные технологии 10](#_Toc5027)

[3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ 11](#_Toc25821)

[3.3.1 Класс DrawingArea 11](#_Toc27136)

[3.3.2 Класс MainWindow 13](#_Toc30417)

[3.3.3 Класс Shape 14](#_Toc10143)

[4. РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ 19](#_Toc19363)

[4.2 Программный модуль рисования 19](#_Toc18368)

[4.4 Реализация модуля FiltraringCommand 19](#_Toc14486)

[5 РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ 21](#_Toc7550)

[ЗАКЛЮЧЕНИЕ 24](#_Toc21436)

[ЛИТЕРАТУРА 25](#_Toc8339)

[ПРИЛОЖЕНИЕ A 26](#_Toc20489)

[ПРИЛОЖЕНИЕ Б 27](#_Toc4517)

[ПРИЛОЖЕНИЕ В 28](#_Toc26377)

[ПРИЛОЖЕНИЕ Г 29](#_Toc14753)

[ПРИЛОЖЕНИЕ Д 30](#_Toc9535)

**ВВЕДЕНИЕ**

Сегодня работа с графикой встречается практически повсеместно: ей занимаются как профессиональные фотографы, так и новички-любители. Соответственно и графические редакторы так же бывают разными : от масштабных проектов по многоплановой обработке и редактированию изображений, до простейшего рисования. В рамках курсового проекта мной была выбрана тема графического редактора, способного выполнять базовые операции редактирования графики, как у его аналогов, однако, имеющего ряд дополнительных функций, не реализованных у конкурентов.

В основе курсового проекта лежит идея создания простого и небольшого редактора, который был бы интуитивно понятен пользователям, и при этом имел расширенный функционал.

Приложение рассчитано в первую очередь на рядовых пользователей, не нуждающихся в профессиональной обработке изображений, но заинтересованных в их редактировании на уровне изменения формата, рисования, удаления шумов, коррекции гаммы и т.д. Уже на данной стадии разработки приложение способно удовлетворять базовым потребностям при редактировании, а в перспективе, при расширении функционала, сможет стать достойным представителем данной сферы на рынке.

Польза от разработки данного ПО заключается в изучении основ работы с изображениями, способов рисования различных форм, а также алгоритмов, реализующих виды редактирования, такие как заливку области, основанные на алгоритмах обхода графов. Были изучены алгоритмы удаления шумов при анализе сигналов, а так же методы кодирования изображений в различных форматах.

1. **ПОСТАНОВКА ЗАДАЧ**

Цель работы:

Разработать графический редактор для работы с растровой графикой, обеспечивающий пользователю удобные инструменты для создания и редактирования изображений. Программа должна предоставлять широкие возможности по работе с растровыми изображениями, а также обладать интуитивно понятным интерфейсом для комфортной работы.

Задачи работы:

1. Изучить существующие графические редакторы для работы с растровой графикой, выявить их основные функциональности и особенности интерфейса.

2. Изучить основные принципы работы с растровыми изображениями, включая форматы файлов, цветовые пространства, инструменты редактирования и техники обработки изображений.

3. Разработать пользовательский интерфейс графического редактора, включающий в себя:

- Набор инструментов для создания и редактирования растровых изображений;

- Инструменты для настройки цветовой палитры и применения эффектов к изображению.

4. Реализовать алгоритмы для обработки растровых изображений, включая операции изменения размера, медианная фильтрация, коррекции цветов и т.д.

5. Провести тестирование программы на различных типах растровых изображений, включая создание, редактирование и применение различных эффектов.

6. Подготовить отчет по курсовой работе, включающий описание основных функциональностей разработанного графического редактора и результаты тестирования.

Ожидаемые результаты:

В результате выполнения данной курсовой работы предполагается создание графического редактора, который обеспечит пользователю удобные инструменты для работы с растровыми изображениями. Программа должна быть функциональной, обладать широкими возможностями по редактированию изображений и обеспечивать высокую производительность при работе с большими файлами.

**1.1 Основные термины**

Пиксель – наименьший логический элемент двумерного цифрового изображения в растровой графике, или элемент матрицы дисплеев, формирующих изображение. Пиксель представляет собой неделимый объект прямоугольной или круглой формы, характеризуемый определённым цветом.

Растр – сетка пикселей, обладающих следующими важными свойствами: размер изображения в пикселях (его ширина и высота), количеством используемых цветов или глубиной цвета, цветовой моделью (например RGB) и разрешение.

RGB (аббревиатура от английского red, green, blue) – [аддитивная](https://ru.wikipedia.org/wiki/%D0%90%D0%B4%D0%B4%D0%B8%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D1%81%D0%BC%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_%D1%86%D0%B2%D0%B5%D1%82%D0%BE%D0%B2" \o "Аддитивное смешение цветов) [цветовая модель](https://ru.wikipedia.org/wiki/%D0%A6%D0%B2%D0%B5%D1%82%D0%BE%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C" \o "Цветовая модель), описывающая способ кодирования [цвета](https://ru.wikipedia.org/wiki/%D0%A6%D0%B2%D0%B5%D1%82" \o "Цвет) для [цветовоспроизведения](https://ru.wikipedia.org/wiki/%D0%A6%D0%B2%D0%B5%D1%82%D0%BE%D0%B2%D0%BE%D1%81%D0%BF%D1%80%D0%BE%D0%B8%D0%B7%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5" \o "Цветовоспроизведение). Выбор основных цветов обусловлен особенностями физиологии восприятия цвета [сетчаткой](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%B1%D0%BE%D1%87%D0%BA%D0%B8_(%D1%81%D0%B5%D1%82%D1%87%D0%B0%D1%82%D0%BA%D0%B0)" \o "Колбочки (сетчатка)) человеческого глаза.

Медианный фильтр – один из видов [цифровых фильтров](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%BE%D0%B9_%D1%84%D0%B8%D0%BB%D1%8C%D1%82%D1%80" \o "Цифровой фильтр), широко используемый в [цифровой обработке сигналов](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%BE%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D1%81%D0%B8%D0%B3%D0%BD%D0%B0%D0%BB%D0%BE%D0%B2" \o "Цифровая обработка сигналов) и [изображений](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%BE%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%B8%D0%B7%D0%BE%D0%B1%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9" \o "Цифровая обработка изображений) для уменьшения уровня [шума](https://ru.wikipedia.org/wiki/%D0%A8%D1%83%D0%BC" \o "Шум). Значения отсчётов внутри окна фильтра сортируются в порядке возрастания (убывания); и значение, находящееся в середине упорядоченного списка, поступает на выход фильтра. В случае чётного числа отсчётов в окне выходное значение фильтра равно среднему значению двух отсчётов в середине упорядоченного списка. Окно перемещается вдоль фильтруемого сигнала и вычисления повторяются. Медианная фильтрация – эффективная процедура обработки сигналов, подверженных воздействию импульсных помех.

Гамма-коррекция – предыскажения [яркости](https://ru.wikipedia.org/wiki/%D0%AF%D1%80%D0%BA%D0%BE%D1%81%D1%82%D1%8C" \o "Яркость) чёрно-белого или [цветоделённых](https://ru.wikipedia.org/wiki/%D0%A6%D0%B2%D0%B5%D1%82%D0%BE%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5" \o "Цветоделение) составляющих цветного [изображения](https://ru.wikipedia.org/wiki/%D0%98%D0%B7%D0%BE%D0%B1%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5" \o "Изображение) при его записи в [цифровой фотографии](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D1%84%D0%BE%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F" \o "Цифровая фотография).

Апертура – прямоугольная область вокруг текущего [пикселя](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D0%BA%D1%81%D0%B5%D0%BB" \o "Пиксел), участвующая в преобразовании.

**2 ОБЗОР ЛИТЕРАТУРЫ**

**2.1 Обзор аналогов**

В сфере работы с графикой существует множество приложений, обеспечивающих редактирование изображений : Paint for Windows, PaintBrush for Mac, adobe photoshop и многие другие.

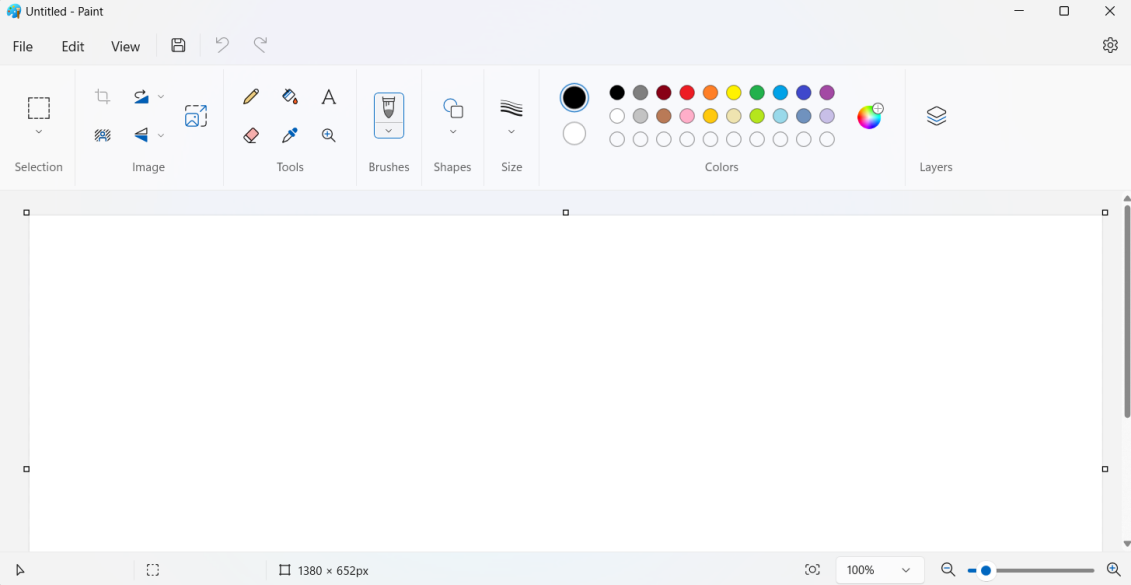


Рисунок 2.1.1 - Paint for Windows

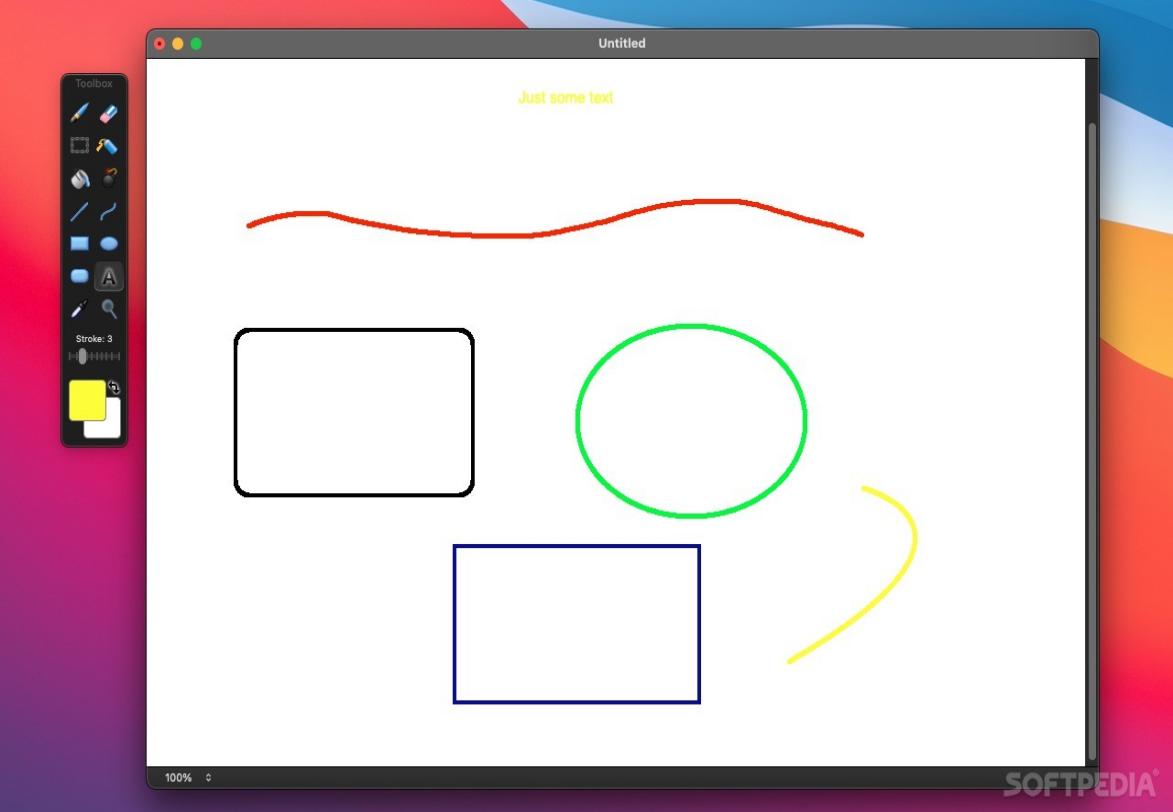


Рисунок 2.1.2 - PaintBrush for Mac

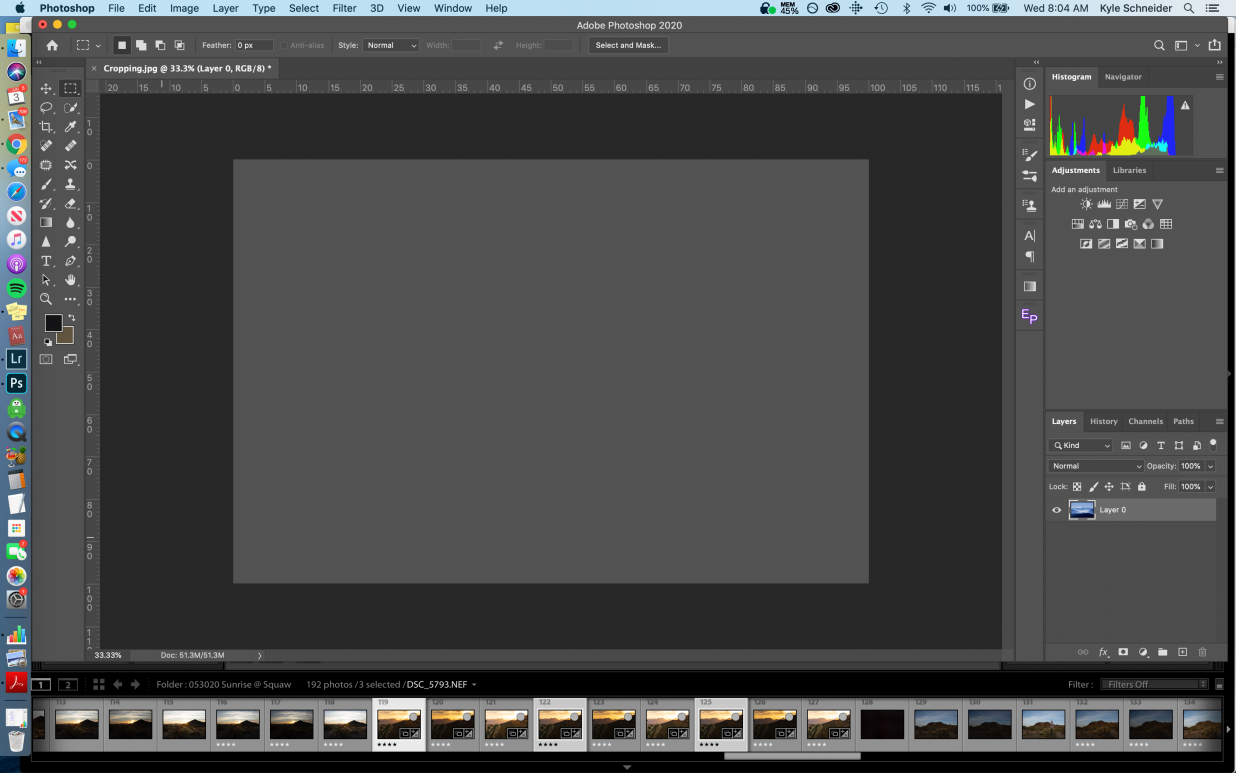


Рисунок 2.1.3 - adobe photoshop

Все эти приложения обладают своими уникальными свойствами, и поэтому, для того, чтобы сделать данный проект приближенный к перечисленным, необходимо в первую очередь реализовать функционал, объединяющий уже существующие аналоги. Для этого выделим основные возможности конкурентов:

1. Поддержка многих форматов изображений.
2. Поддержка отмены последних действий.
3. Возможность изменения размеров изображений.
4. Рисование простейших фигур.

**2.2 Функциональные требования**

После анализа аналогов можно сформулировать основные требования, которым должен удовлетворять проект:

1. Поддержка многих форматов изображений.
2. Рисование фигур.
3. Сохранение изображений.
4. Изменение размеров изображений
5. Возможность отката последних действий
6. Возможность отражения изображения.

Дополнительные требования (фильтрация изображения):

1. Медианная фильтрация изображения
2. Коррекция гаммы изображения
3. Реализация собственной версии заливки области изображения

**2.3 Основные технологии**

Для написания курсового проекта был выбран язык программирования С++. Это компилируемый, мультипарадигмальный язык общего назначения с поддержкой процедурного программирования (возможность реализации на чистом С), обобщенного программирования (шаблоны) и объектно-ориентированного (классы и их объекты), последнее из которых и лежит в основе данного курсового проекта. С++ широко используется для разработки программного обеспечения, являясь при этом одним из самых популярных я зыков программирования в мире. На этом языке можно разработать как простейшие консольные приложения, так и целые операционные системы и драйверы.­­­­­­­

Разработка курсового проекта строилась на основе парадигмы объектно-ориентированного программирования, которая подразумевает представление всей программы в виде совокупности взаимодействующих между собой объектов, которые являющихся экземплярами некоторых абстрактных сущностей – классов. Сами классы в свою очередь обладают собственными функциями, называемыми методами, которые их реализую.

Широко использовались возможности языка С++, такие как работа с указателями и динамической памятью, хранение объектов в стандартных контейнерах библиотеки STL, использование абстрактных классов для доступа к классам потомкам с помощью полиморфизма.

Создание проекта выполнялось в кроссплатформенном фреймфорке Qt версии 6.6. Данный фреймворк позволяет упростить работу с реализацией графического интерфейсом за счет наличия метаобъектного компилятора, обеспечивающего механизм сигналов/слотов, когда одни классы способны испускать сигналы при определенных условиях(например при взаимодействии с пользователем), а другие перехватывать их и выполнять некоторые предопределенные операции (слоты).

**3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

**3.1 Структура входных и выходных данных**

В данной работе входными данными является файл,в котором находятся иконки инструментов и не только для программы.

**3.2 Разработка диагриммы классов**

В ходе работы была разработана диаграмма классов представленная в

приложении А

**3.3 Описание классов**

**3.3.1 Класс DrawingArea**

Класс DrawingArea – это основной класс программы,, который представляет собой виджет рабочей зоны. Именно на нем происходит все рисование а также редактирование: Его полями являются:

1. QUndoStack \*\_undoStack – стек команд, реализующих DrawingArea.
2. QImage \_image – указатель на изображение.
3. int \_penWidth – ширина ручки.
4. QPoint lastPoint – последняя позиция мыши.
5. QColor \_penColor – цвет пера.
6. QColor \_oldPenColor – цвет ручки, использующаяся для запоминания старого цвета.
7. std::string \_currentShape – строка указывающая на фигуру.
8. bool isDrawing – поле ,отвечающая за “рисование”.
9. QImage \_afterDrawingImage – указатель на изображение ,использующаяся для redo/undo методах.
10. bool \_isEraser – поле, которое указывает на ластик.

Методы класса Drawing area:

1. void setCreateEllipse()– слот, задающий создание эллипса.
2. Void setCreateRectangle()– слот, задающий создание прямоугольника.
3. void setCreatePen()– слот, задающий создание ручки.
4. void setCreateColorPicker()– слот, задающий создание пипетки.
5. void setCreateEraser()– слот, задающий создание ластика.
6. void setCreateFilledShape()– слот, задающий создание заливки.
7. Void *mousePressEvent*(QMouseEvent \*event) – перегруженное события нажатия клавиши мышки, которое отвечает за начало рисования
8. void *mouseReleaseEvent*(QMouseEvent \*event) – перегруженное события движения мыши, которое отвечает за изменение фигур при рисовании
9. void mouseMoveEvent(QMouseEvent \*event) – перегруженное событие отпускания кнопки мышки, которое создает объект класса DrawCommand и заносит его в \_undoStack.
10. void medianFilter(const int ratio) – метод, который создает объект класса FiltratingCommand и добавляет его в undoStack.
11. void gammaCorrection(const double gamma) – метод, который создает объект класса CorrectionCommand и добавляет его в undoStack.
12. void resizeIamge(const QSize newSize) – метод, который создает объект класса ResizeCommand и добавляет его в undoStack.
13. void flip(bool horizontal, bool vertical) - метод, который создает объект класса FlipCommand и добавляет его в undoStack.
14. bool isModified() - метод, который указывает было ли изменено изображение.
15. void createNewImage() - метод для создания нового изображения.
16. bool openImage(const QString &fileName) - метод открытия файла.
17. Bool saveImage(const QString &fileName, const char \*fileFormat) - метод сохранения изображения.
18. void setPenColor(const QColor &newColor) - установка цвета пера.
19. void setPenWidth(int newWidth) - установка ширины пера.
20. QColor getPenColor() - возвращает цвет пера.
21. int getPenWidth() - возвращает ширину пера.
22. QSize getImageSize() - возвращает размер изображения.
23. void toggleEraserMode() - переключение между ластиком.
24. void drawEllipses(QMouseEvent \*event) - рисование элипса.
25. void drawRectangle(QMouseEvent \*event) - рисование прямоугольника.
26. void ColorPicker(QMouseEvent \*event) - метод использования пипетки.
27. void fillShape(QMouseEvent \*event) - метод заливки.
28. void paintEvent(QPaintEvent \*event) - инициализация рисования.

Таким образом классы команд реализуют класс DrawingArea, чьи методы завязаны на них. Сам же класс DrawingArea управляет этими командами, т.е. вызывает их в случае необходимости, совершает откат или повторение.

**3.3.2 Класс MainWindow**

Класс MainWindow– это класс, представляющий собой графический интерфейс программы: именно через него идет большая часть взаимодействия программы с пользователем. Данный класс реализует все кнопки приложения, панель инструментов и меню.Основные Поля MainWindow являются:

1. QMenu \*\_saveAsMenu - меню сохранения изображения.
2. QMenu \*\_fileMenu - меню открытия изображения.
3. QMenu \*\_doRedoMenu - меню отмены действия.
4. QMenu \*\_brushMenu - меню инструментов.
5. QMenu \*\_effectsMenu - меню эффектов.
6. QMenu \*\_addOptionsMenu - меню изменения изображения.
7. QMenu \*\_helpMenu - меню помощи.

Основные Методы:

1. void filtrate() – слот, являющийся графической оберткой метода void medianFilter(const int ratio) класса DrawingArea для взаимодействия с пользователем.
2. void correction() – слот, являющийся графической оберткой метода void gammaCorrection(const double gamma) класса DrawingArea для взаимодействия с пользователем.
3. void changeSize() – слот, являющийся графической оберткой метода void resizeIamge(const QSize newSize) класса DrawingArea для взаимодействия с пользователем.
4. void create DrawActiongroup()– метод создания действий, связанных с рисованием. Кнопки этих действий подключаются к соответствующим слотом класса DrawingArea.
5. void createActions()– метод создания действий.
6. QToolBar\* createToolBar()– метод создания панели инструментов.
7. void createMenues() – метод создания меню.

Таким образом, класс MainWindow, при взаимодействии с пользователем лишь перенаправляет все команды в «управляющий модуль», представленный классом DrawingArea.

**3.3.3 Класс Shape**

Класс Shape– это абстрактный класс, который в контексте данной программы используется для хранения объектов производных классов. Использование полиморфизма позволяет значительно упростить процесс рисования и сделать его шаблонным для любых форм. Основными полями Shapeявляются:

1. int \_penWidth - ширина пера.
2. QColor \_penColor - цвет пера.
3. QImage\* \_image - указатель на изображение.

Основные методы:

1. virtual void *draw*(QPainter &painter) = 0 – чисто виртуальный метод, отвечающий за нанесение фигур на изображение. Переопределяется в производных классах фигур.
2. virtual void *update*(const QPoint &toPoint) = 0 – чисто виртуальный метод, отвечающий за то, чтобы размер прямоугольника фигуры и его содержимое были актуальны. Переопределяется в производных классах фигур.
3. virtual QRect *rect*() = 0 – чисто виртуальный метод, отвечающий за нахождение прямоугольника, содержащего фигуру. Переопределяется в производных классах фигур.

От класса Shape наследуется множество фигур, однако они лишь являются частными случаями и узкими специализациями широкого функционала рисования этого класса. Стоит отметить ,что для упрощения работы конструкторы классов наследников принимают одинаковый список параметров для инициализации.

**3.3.4 Класс Ellipse**

Класс Ellipse – это производный класс от класса Shape,используемый для хранения и рисования элипса.Основными полями Ellipse является:

QRect \_rectangle - указатель на фигуру.

Основные методы:

1. void draw(QPainter &painter) - метод для нанесения фигуры на изображения.
2. void update(const QPoint &toPoint) - метод для изображения фигуры.
3. QRect rect() - метод для нахождения прямоугольника фигуры.

**3.3.5 Класс Rectangle**

Класс Rectangle – это производный класс от класса Shape,используемый для хранения и рисования элипса.Основными полями Rectangle является:

QRect \_rectangle - указатель на фигуру.

Основные методы:

1. void draw(QPainter &painter) - метод для нанесения фигуры на изображения.
2. void update(const QPoint &toPoint) - метод для изображения фигуры.
3. QRect rect() - метод для нахождения прямоугольника фигуры.

**3.3.6 Класс Filledshape**

Класс Filledshape - ‘это производный класс от класса shape ,используемый для заполнения фигур и не только.Основными полями Filledshape являются:

QRect \_rectangle - указатель на фигуру.

std::vector<QPoint> \_points - контейнер для хранения пикселей.

Основные методы:

1. void draw(QPainter &painter) - метод для нанесения фигуры на изображения.
2. void update(const QPoint &toPoint) - метод для изображения фигуры.
3. QRect rect() - метод для нахождения прямоугольника фигуры.

**3.3.7 Класс Drawcommand**

Класс Drawcommand **-** это класс для команд отмены рисования.основными полями Drawcommand являются:

1. QImage \*\_image - указатель на изображение.
2. \_afterDrawingImage - поле для undo команды.
3. \_beforeDrawingImage - поле для redo команды.
4. QWidget \*\_drawingArea - указатель на поле для рисования.

Основные методы:

1. void undo() - команда отмены действия.
2. void redo() - обратная команда отмены действия.

**3.3.8 Класс Resizecommand**

Класс resizecommand **-** это класс для команд отмены изменения размера изображения.основными полями resizecommand являются:

1. QWidget \*\_drawingArea - указатель на поле для рисования.
2. QImage \*\_image - указатель на изображение.
3. QImage \_undoImage - указатель на изображение(для undo команды).
4. QSize \_oldSize - старый размер изображения.
5. QSize \_newSize - новый размер изображения.

Основные методы:

1. void undo() - команда отмены действия.
2. void redo() - обратная команда отмены действия.

**3.3.9 Класс Correctioncommand**

Класс Correctioncommand **-** это класс для команд отмены изменения размера изображения.основными полями Correctioncommand являются:

1. QWidget \*\_drawingArea - указатель на поле для рисования.
2. QImage \*\_image - указатель на изображение.
3. QImage \_undoImage - указатель на изображение(для undo команды).
4. double \_gamma - значения для гамма коррекции.

Основные методы:

1. void undo() - команда отмены действия.
2. void redo() - обратная команда отмены действия.
3. void adjust() - гамма коррекция.

**3.3.10 Класс filtraringcommand**

Класс filtraringcommand**-** это класс для команд отмены изменения размера изображения.основными полями filtraringcommand являются:

1. QWidget \*\_drawingArea - указатель на поле для рисования.
2. QImage \*\_image - указатель на изображение.
3. QImage \_undoImage - указатель на изображение(для undo команды).
4. int \_ratio - значения для медианной фильтрации.

Основные методы:

1. void undo() - команда отмены действия.
2. void redo() - обратная команда отмены действия.
3. void filtrate() - медианная фильтрация.

**3.3.11 Класс flipcommand**

Класс flipcommand **-** это класс для команд отмены изменения размера изображения.основными полями flipcommand являются:

1. QWidget \*\_drawingArea - указатель на поле для рисования.
2. QImage \*\_image - указатель на изображение.
3. bool \_horizontal - поле используемое для определения как будет изменено изображение.
4. bool \_vertical - поле используемое для определения как будет изменено изображение.

Основные методы:

1. void undo() - команда отмены действия.
2. void redo() - обратная команда отмены действия.

**4. РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ**

**4.1 Разработка схем алгоритмов(два наиболее важных метода)**

Схемы алгоритмов gammaCorection и FilledShape представлены в преложении Б и В соответственно.

**4.2 Программный модуль рисования**

1.Начало

2. Пользователь выбирает текущую операцию рисования, такую как фигура, заливка и т.д.

3. При выборе фигуры срабатывает один из слотов-инициализаторов функции создания фигуры в классе DrawingArea.

4. Каждый слот инициализирует указатель на функцию создания фигуры, которая содержится в классе DrawingArea.

5. Когда пользователь нажимает мышью в рабочей зоне, срабатывает событие mousePressEvent, создающее фигуру.

6. При движении мышью используется метод drawMouseLine для рисования.

7. Когда пользователь прекращает рисовать и отпускает клавишу, срабатывает событие mouseReleaseEvent, которое передает команду рисования в стек команд и рисует выбранную фигуру.

8. Итоговая фигура переносится на изображение в зависимости от типа фигуры с помощью чисто виртуального метода draw(), переопределенного в классах-наследниках Shape.

9. Для отката используется метод undo() класса DrawCommand, а для отметы отката - метод redo().

10.Конец

**4.4 Реализация модуля FiltraringCommand**

1.Начало

2. Инициализация размера апертуры (в данном случае 3x3) и определение центрального пикселя апертуры.

3. Обход каждого пикселя изображения во внутреннем цикле, начиная с первого пикселя (1,1) и заканчивая предпоследним пикселем (width-1, height-1), где width - ширина изображения, height - высота изображения.

4. Для каждого пикселя изображения формируется апертура, включающая его и его окрестность. Если пиксель находится на краю изображения, апертура строится только из доступных пикселей.

5. Для каждой составляющей цвета (красный, зеленый, синий) создаются массивы для хранения значений.

6. Значения цветов каждого пикселя из апертуры заносятся в соответствующие массивы.

7. Для каждой составляющей цвета вычисляется среднее значение по массиву значений.

8. Полученные средние значения комбинируются вместе для формирования нового цвета пикселя.

9. Новый цвет присваивается центральному пикселю апертуры.

10. Процесс повторяется для всех пикселей изображения, за исключением краевых пикселей, для которых апертура строится из меньшего количества пикселей.

11. В результате получается изображение, на котором шум уменьшен за счет сглаживания цветов в окрестности каждого пикселя.

12.Конец

**5 РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ**

При первом запуске программы перед вами появится данное окно (см. рисунок 5.1)

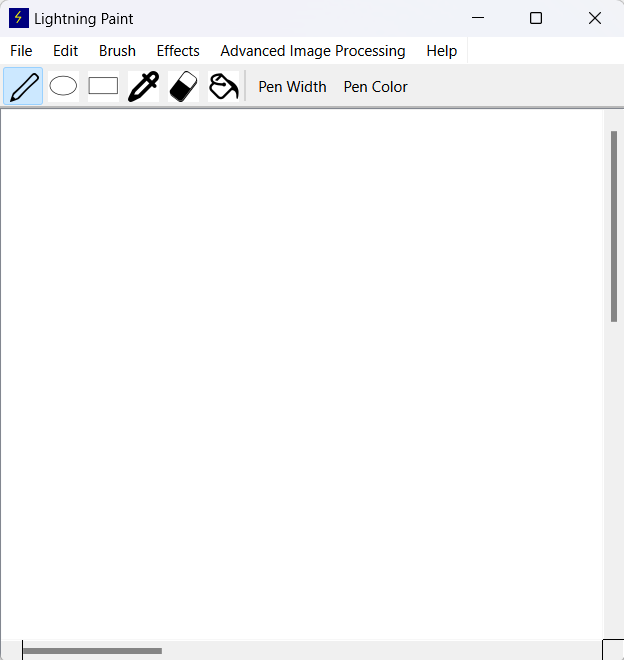


Рисунок 5.1 – Главное окно приложения

Чтобы изменить текущую операцию рисования или параметры ручки вы можете либо воспользоваться панелью инструментов (см. рисунок 5.2), либо открыть вкладку меню Brush и выбрать нужную вам операцию (см. рисунок 5.3).

Чтобы повернуть или отразить изображение воспользуйтесь вкладкой Effects и выберете нужную операцию (см. рисунок 5.4).

Чтобы изменить размер изображения, отфильтровать его или изменить его гамму, откройте вкладку Advanced Image Processing и выберите нужную операцию (см. рисунок 5.5).



Рисунок 5.2 – Панель инструментов

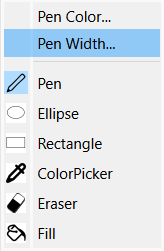


Рисунок 5.3 – Меню настройки кисти



Рисунок 5.4 – Меню эффектов

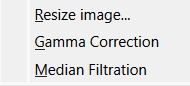


Рисунок 5.5 – Меню расширенных возможностей обработки

Чтобы открыть, сохранить или создать новый файл откройте вкладку меню File и выберите нужную операцию (см. рисунок 5.6).

Для того, чтобы отменить последнее действие вы можете нажать комбинацию CTRL–Z или открыть вкладку Edit и воспользоваться кнопкой Undo.

Для того, чтобы повторить последнее отмененное действие вы можете нажать комбинацию CTRL–Y или открыть вкладку Edit и воспользоваться кнопкой Redo.

Для получения помощи откройте вкладку Help и нажмите кнопку About во всплывающем окне.

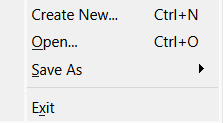


Рисунок 5.6 – Меню работы с файлами

**ЗАКЛЮЧЕНИЕ**

В результате проделанной работы был разработан графический редактор, удовлетворяющий условиям, описанным в разделе 1. Проект был отлажен, протестирован и задокументирован.

Полученный графический редактор является полноценным приложением, которое можно переносить на устройства, работающие на ОС Windows версии 11.

Были получены знания о том, какие существуют подходы к созданию ПО, и как заниматься разработкой последнего максимально эффективно.

В процессе разработке также были углублены знания языка С++ и его расширения для фреймворка Qt Creator, что является очень важным и существенным опытом, на который можно будет опираться в дальнейшем.

Благодаря соблюдению стандартов написания кода и хорошо построенной логике программы ее дальнейшее расширение не должно вызывать особых проблем.

Итоговый продукт получился нетребовательным к аппаратному обеспечению, однако существует исключительная ситуация, которая была ограничена искусственно: высокие затраты памяти и времени при больших объемах зоны заливки. Однако в будущем планируется оптимизировать этот алгоритм так, чтобы надобность в выбросе исключительной ситуации при загрузке большого изображения отпала.

В перспективе можно реализовать:

1. выделение области для обрезания или масштабирования,

2.добавить возможность рисовать правильные фигуры по типу квадрата, круга и т.д,

3.реализовать возможность вставки пользовательского текста,

4.добавить возможность наложения изображений друг на друга с помощью альфа-канала.

**ЛИТЕРАТУРА**

1. Paint for Windows [Электронный ресурс] .– Режим доступа: <https://www.microsoft.com/en-us/windows/paint>
2. Paint for Mac [Электронный ресурс] .– Режим доступа: <https://apps.apple.com/us/app/paint-s/id736473980?mt=12>
3. Adobe photoshop [Электронный ресурс] .– Режим доступа: https://www.adobe.com/ru/products/photoshop.html
4. Шилдт Г. С++ базовый курс / Г. Шилдт - 3-е изд. М. : Вильямс , 2007. – 624с.
5. Страуструп Б. Язык программирования С++ : специальное издание / Б. Страуструп : пер. с англ. – Москва : Бином , 2006.– 1104с.
6. Конструирование программ и языки программирования: метод. Указания по К65 курсовому проектированию для студ. I-40 02 01 “Вычислительные машины, системы и сети” для всех форм обуч. / сост. А. В. Бушкевич , А. М. Ковальчук , И. В. Лукьянова. - Минск: БГУИР , 2009. -320с.
7. Объектно-ориентированное программирование на языке С++:учеб. Пособие /Ю. А. Луцик , В. Н. Комличенко. – Минск: БГУИР , 2008. - 143с.

# **ПРИЛОЖЕНИЕ A**

**(**обязательное**)**

Диаграмма классов

# **ПРИЛОЖЕНИЕ Б**

**(**обязательное**)**

Схема алгоритма. gammaCorection

# **ПРИЛОЖЕНИЕ В**

**(**обязательное**)**

Схема алгоритма. FilledShape

# **ПРИЛОЖЕНИЕ Г**

**(**обязательное**)**

Исходный текст программы

# **ПРИЛОЖЕНИЕ Д**

**(**обязательное**)**

Ведомость документов