МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

Факультет информационных технологий и компьютерной безопасности

(факультет)

КУРСОВОЙ ПРОЕКТ

по дисциплине «Основы программирования на Python»

Тема «Анализ сравнения двух изображений»

Расчетно-пояснительная записка

Разработал студент   
гр. бИВТ -211М.А.Холодулин

подпись, дата инициалы, фамилия

Руководитель А.Н. Юров

подпись, дата инициалы, фамилия

Нормоконтроллер А.Н. Юров

подпись, дата инициалы, фамилия

Защищен\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 дата

Воронеж 2023

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

Кафедра автоматизированных и вычислительных систем

# ЗАДАНИЕ

на курсовой проект

по дисциплине «Основы программирования на Python» \_

Тема «Анализ сравнения двух изображений»

Студент группы бПО-211, Холодулин Максим Александрович\_\_\_\_\_\_\_\_\_\_\_

Фамилия, имя, отчество

Технические условия: аппаратные средства: ПК на базе процессора Intel Core i3-8100 CPU 3.6GHz, ОЗУ 16 Гб; программные средства: cреда разработки VS Code, Язык Знерщт, ОС Windows 10, Microsoft Office Word 2019

Содержание и объем работы: (графические работы, расчеты и прочее)

17 страниц, 7 рисунков \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Сроки выполнения этапов Рассмотрение теоретических сведений (ноябрь 2023); разработка (ноя. -дек. 2023); оформление пояснительной записки (декабрь 2023)

Срок защиты курсового проекта \_\_\_\_\_\_\_\_\_\_декабрь 2023\_\_\_\_\_\_\_\_\_\_

Руководитель А.Н. Юров

подпись, дата инициалы, фамилия

Задание принял студентМ.А.Холодулин

подпись, дата инициалы, фамилия

Воронеж 2023

# Замечания руководителя

Содержание

[Задание на курсовой проект 2](#_Toc122020445)

[Замечания руководителя 3](#_Toc122020446)

[Введение 5](#_Toc122020447)

[1 Теоретическая часть 6](#_Toc122020448)

[1.1 Электронные очереди 6](#_Toc122020449)

[1.3 Используемые инструменты Qt creator 7](#_Toc122020450)

[1.4 Преимущества и недостатки Qt creator 8](#_Toc122020451)

[2 Практическая часть 9](#_Toc122020452)

[2.1 Задание для курсового проектирования 9](#_Toc122020453)

[2.2 Ход выполнения работы 9](#_Toc122020454)

[2.3 Разработка программного средства 11](#_Toc122020455)

[2.4 Тестирование 14](#_Toc122020456)

[2.5 Руководство пользователя 15](#_Toc122020457)

[Заключение 16](#_Toc122020458)

[Список литературы 17](#_Toc122020459)

# Введение

В современном информационном обществе обработка и анализ изображений становятся всё более важными аспектами, находя применение в различных областях, от медицинской диагностики до компьютерного зрения. Программы, способные анализировать и сравнивать изображения, становятся ключевым инструментом для выявления изменений, распознавания образов и поддержания безопасности.

Цель данного проекта заключается в разработке приложения на языке программирования Python, способного проводить анализ и сравнение двух изображений. Основной задачей является создание удобного инструмента, который позволит пользователям быстро и точно определить степень схожести между изображениями.

В контексте анализа сравнения изображений, проект фокусируется на использовании современных методов обработки изображений и библиотек, таких как OpenCV и PySide6, для разработки функционального и интуитивно понятного приложения. Решения, представленные в данном проекте, могут иметь практическое применение в различных областях, включая медицину, безопасность, искусство и многие другие.

# 1 Основы анализа изображений

## 1.1 Понятие цифрового изображения

Цифровое изображение представляет собой дискретное численное представление визуальной информации. Вместо использования непрерывных значений, как в аналоговых изображениях, цифровые изображения состоят из множества дискретных элементов, называемых пикселями. Каждый пиксель представляет собой точку на изображении и имеет свои числовые значения, обозначающие интенсивность или цвет данной точки.

## 1.2 Характеристики цифрового изображения

Основные характеристики цифрового изображения определяются его структурой и характеристиками пикселей.

Важными параметрами являются:

Разрешение (Resolution) определяет количество пикселей вдоль каждого измерения. Оно измеряется в ширина x высота пикселей. Чем выше разрешение, тем более детализированным будет изображение.

Глубина цвета (Color Depth) определяет количество битов, используемых для представления цвета каждого пикселя. Например, изображение с глубиной цвета 8 бит может представлять 256 различных цветов, тогда как изображение с глубиной цвета 24 бита может представлять более 16 миллионов цветов.

Формат файла (File Format) определяет, как информация о изображении сохраняется и организуется. Популярные форматы включают JPEG, PNG, BMP и другие. Каждый формат имеет свои преимущества и недостатки в зависимости от конкретного применения.

Каналы цвета (Color Channels) обычно представлены в виде комбинации базовых цветов — красного, зеленого и синего (RGB). Каждый канал представляет интенсивность соответствующего цвета.

Битовая глубина (Bit Depth) определяет количество битов, используемых для представления интенсивности каждого пикселя. Например, изображение с битовой глубиной 8 бит будет иметь 256 уровней яркости.

1.3 Методы сравнения изображений

Структурные методы сравнения (Structural Similarity Index - SSIM): Измеряют схожесть основываясь на структурных элементах изображения, таких как текстуры и контраст. SSIM предоставляет нормализованный коэффициент, отражающий степень схожести.

Пиксельные методы сравнения оценивают различия между интенсивностью пикселей в изображениях. Простые, но эффективные методы.

Функциональные методы сравнения основаны на анализе функций, таких как гистограммы распределения интенсивности.

### 1.3.1 Структурные методы сравнения

Structural Similarity Index (SSIM) измеряет схожесть структуры изображений, учитывая восприятие человеческого зрения. Выдает значение от -1 до 1, где 1 указывает на полное сходство. Широко используется в задачах, где важна структурная информация, таких как медицинская диагностика и обработка изображений.

Mean Structural Similarity Index (MSSIM). Улучшенная версия SSIM, оценивающая среднюю структурную схожесть между несколькими изображениями. Эффективен в оценке качества сжатых изображений и видео.

### 1.3.2 Пиксельные методы сравнения

Mean Squared Error (MSE). Среднеквадратичная ошибка измеряет среднеквадратичное отклонение интенсивности пикселей между двумя изображениями. Меньшие значения указывают на большую схожесть. Прост в реализации, но чувствителен к яркостным изменениям.

Structural Dissimilarity Index (DSSIM). Расширение MSE, учитывающее структурные различия. Использует веса для разных частей изображения. Эффективен в задачах, где важны как локальные, так и глобальные структурные изменения.

### 1.3.3 Функциональные методы сравнения

Гистограмма распределения интенсивности. Сравнение распределения интенсивности пикселей между изображениями. Может быть использовано для выявления цветовых изменений. Широко используется в задачах, связанных с анализом цвета и распределения яркости.

Корреляция. Измерение степени линейной зависимости между интенсивностью пикселей двух изображений. 1 указывает на полное сходство. Подходит для выявления смещений и изменений в положении объектов.

# 2 Практическая часть

## 2.1 Задание для курсового проектирования

Разработать программное средство «Очередь пациентов» с графическим пользовательским интерфейсом, используя Qt/С++.

Приложение должно реализовывать электронную очередь пациентов, выводить на экран список пациентов в очереди. Пациент вносится в очередь с присвоением уникального номера в очереди. По нажатию на кнопку «Прием» пациент изымается из очереди.

## 2.2 Ход выполнения работы

Программа, которая была разработана в ходе курсового проектирования содержит три основных части:

* mainWidget;
* showQueue;
* widgetScreen.

В каждой части в свою очередь содержится по три файла с разными расширениями:

* заголовочный файл .cpp;
* файл реализации .cpp;
* файл интерфейса .ui.

Проще всего рассмотреть принцип работы программы можно с помощью схем. Схема алгоритма работы программы представлена на рисунке 1. Схема классов показана на рисунке 2.

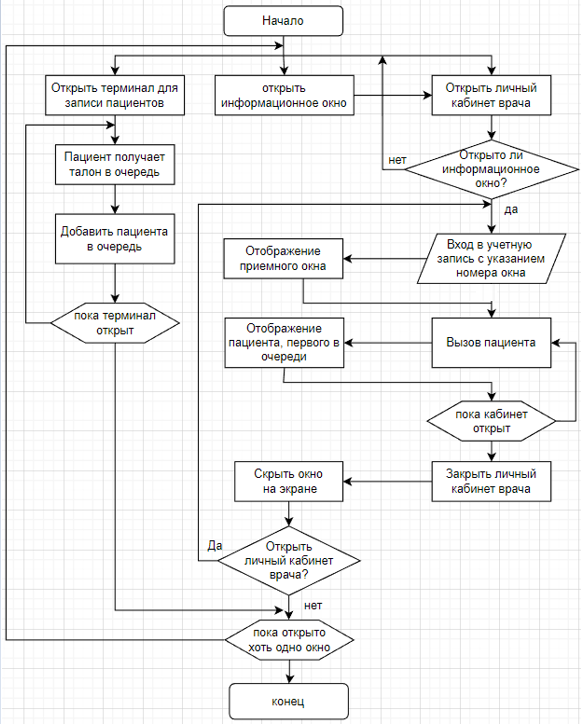


Рисунок 1 - Схема алгоритма работы программы

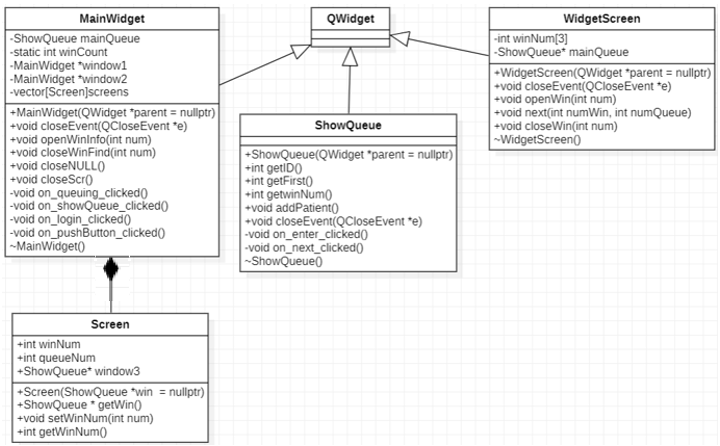


Рисунок 2 – Схема классов

## 2.3 Разработка программного средства

Поскольку в программе присутствует 4 окна, я решил создавать пользовательский интерфейс с помощью Qt Designer, т.к. это ускорит процесс разработки. Первый графический интерфейс(рисунок 3 – рисунок4) состоит из двух располагается в одном файле mainWidget.ui, поскольку для этих окон нужно меньше всего элементов. Здесь содержится два GroupBox-а для того, чтобы было удобно скрывать ту или иную панель. GreedLayot позволяет растягивать и сжимать содержимое при изменении размера окна.

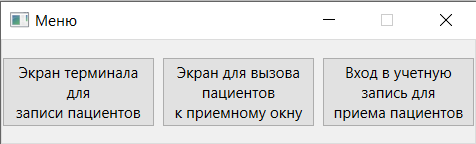


Рисунок 3 – Главное окно

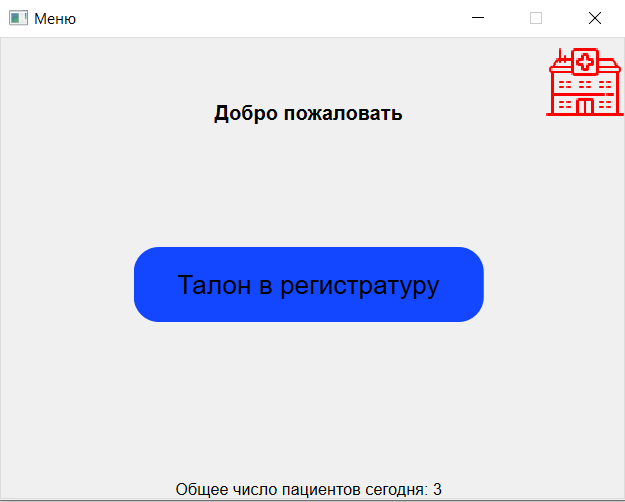


Рисунок 4 – Терминал записи в регистратуру

В основе программы лежит библиотека STL - queue. При нажатии на первую кнопку меню, открывается терминал для записи в очередь отдельным окном. При нажатии на кнопку «Талон в регистратуру», вызывается метод ShowQueue::addPatient(). В классе ShowQueue присутствует статическая переменная patID, которая увеличивается на единицу и добавляется в очередь static std::queue<int> patientQueue. Очередь статическая, чтобы она была одна для всех объектов класса. Окно для приема пациентов содержится в файле showQueue.ui (рисунок 5).

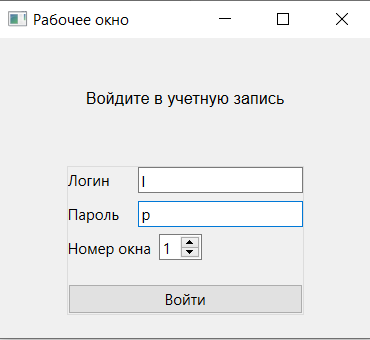


Рисунок 5 – Вход в личный кабинет врача

После удачной авторизации, номер окна отображается на экране widgetScreen.ui (рисунок 6). Можно обратить внимание на то, что при закрытии приемного окна, информация на экране также удаляется (рисунок 7).

Для этого посылаются сигналы:

* emit openWinToMain(winNum);
* emit openWinToScreen(winNum).

Первый сигнал подается на слот в файл mainWidget и добавляет лишь информацию по номеру открытого окна. Второй сигнал ловится в классе WidgetScreen и отвечает за отображение открытого окна. При нажатии на кнопку «Следующий пациент» сначала проверятся не пустая ли очередь. В случае если пустая, выскочит окно с предупреждением. В ином случае, из очереди удалится пациент, находящийся первый в очереди. Но перед удалением, будет подан сигнал emit next(getwinNum(), patientQueue.front()), из которого экран получит информацию о том, какое окно вызывает пациента и номер этого пациента в очереди. В итоге на экране будет вызван новый талон.



Рисунок 6 – Информационное окно

Для того, чтобы обработать случай с закрытием рабочего окна, можно использовать событие. В моем случае я переопределил метод класса QWidget void closeEvent(QCloseEvent \*e). Такой метод нужен для каждого класса в моем проекте, поскольку открытие окна осуществляется с помощью оператора new. А поскольку окно закрылось, то значит пора освобождать выделенную память. Если закрываемое окно принадлежит MainWidget, то в closeEvent() просто освобождаем память. В ином случае, поскольку все окна создавались в файле MainWidget, нужно подавать сигнал о закрытии в этот файл. При закрытии рабочего окна, которое принадлежит классу ShowQueue сигнал должен получить еще и экран «Текущий прием» в файл WidgetScreen. Область, принадлежащая закрытому окну просто скрывается методом hide(). В дальнейшем при открытии нового рабочего окна, операция по включению окну повторится и на экране.

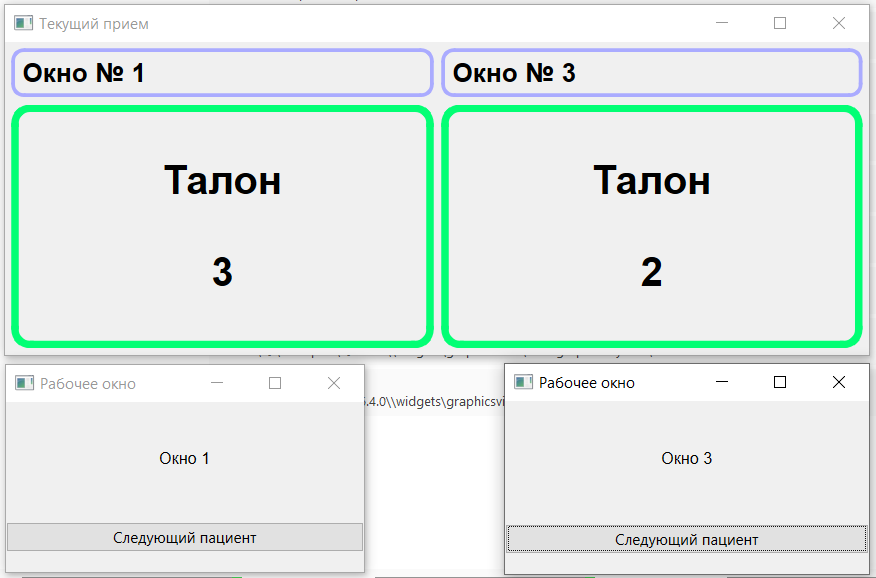


Рисунок 7 – Закрытие рабочего окна

## 2.4 Тестирование

Перед использованием приложения в первый раз, рекомендуется протестировать работоспособность всех опций:

1. Открыть терминал для записи пациентов в очередь;
2. Убедиться, что при нажатии на кнопку «Получить талон», внизу терминала обновляется информация о количестве пациентов в очереди;
3. Проверить, чтобы при попытке входа в учетную запись врача при не включенном экране, высвечивалась информация о необходимости включить экран;
4. После авторизации в рабочем окне удостоверится, что оно высветилось на экране и номер, введенный при авторизации совпадает с отображенным;
5. Убедиться, что нельзя открыть более трёх рабочих окон;
6. При нажатии на кнопку «Следующий пациент», на экране должен появиться/измениться номер вызываемого пациента под соответствующим окном;
7. Удостовериться, что при закрытии рабочего окна, оно исчезает с экрана, а при открытии взамен нового окна, не появляется ошибок.

## 2.5 Руководство пользователя

Скачать программу на каждое устройство:

* терминал для записи;
* информационный экран;
* рабочие компьютеры.

Подключить устройства между собой с помощью кабелей или интернета. При запуске программы включить все устройства. В главном окне программы, открыть поочередно окна для каждого устройства. Экран необходимо подключить заранее. Не рекомендуется открывать окна с одинаковыми номерами, а также закрывать рабочее окно без прохождения авторизации. Возможна одновременная работа с не более чем тремя рабочими окнами. Пользователь может просматривать информацию о количестве пациентов. Пользователь может принимать пациентов только в порядке очереди. Для открытия нового рабочего окна в случае трёх активных окон необходимо закрыть одно из активных окон.

# Заключение

В данной курсовой работе удалось разработать приложение с пользовательским интерфейсом, используя компоненты среды Qt creator.

Оно содержит три различных класса, которые практически не связаны между собой, но благодаря возможности отправки сигналов, становится возможным оказывать влияние на одни классы при действиях пользователя в других классах.

Во время курсового проектирования достигнуты следующие цели:

* создание приложения с пользовательским интерфейсом;
* использование в приложении нескольких активных окон;
* изменение положения виджетов при изменении размера окна;
* образование связи между разными окнами для передачи информации.

Так же в ходе курсового проектирования были получены новые и закреплены уже имеющиеся знания и навыки для работы с Qt creator. А именно:

* проектирование приложения с графическим интерфейсом;
* получение информации из графического интерфейса;
* работа с виджетами;
* взаимодействие различных окон с помощью сигналов.

Исходный код курсового проекта можно посмотреть по ссылке: https://github.com/Kholodulin/VSP/tree/main/patientQueue

# Список литературы