Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Институт «Электронных и информационных систем»

Кафедра «Информационных технология и систем»

Лабораторная работа №1

**«Создание средства просмотра изображений»**

Отчёт по дисциплине:

«Распознавание образов и обработка изображений»

Выполнил студент группы 8091:

\_\_\_\_\_\_\_\_\_\_ / Баранов А. С.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Принял:

\_\_\_\_\_\_\_\_\_\_ / Назаров А. Г.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

**Великий Новгород**

**2022**

**Цель лабораторной работы**

Подготовка рабочего программного окружения для выполнения лабораторных работ.

Разработать и реализовать программу просмотра изображений.

**Программное окружение**

Для выполнения данной лабораторной работы было собрано следующее окружение:

* Язык программирования: C++
* Операционная система: MS Windows 10 LTSC
* Компилятор: MinGW 64-bit
* Среда программирования: QtCreator 5.0.2

**Ограничение пролистывания изображений**

Во время просмотра изображений пользователь может нажать кнопку пролистать влево при просмотре первого изображения в списке, либо нажать пролистать вправо при просмотре последнего изображения в списке. Есть два пути решения: реализовать пролистывание по кругу, показывать окно предупреждения о крайнем положении в списке, либо просто ограничить действие кнопки перелистывания при положении в начале или конце списка.

Мы воспользуемся третьим вариантом. Для этого нужно отключить **prevAction**, когда пользователь находится в начале списка, и отключить **nextAction**, когда пользователь находится в конце списка, т.е. кнопки не будут ничего выводить.

**Добавление значков действий**

Для приятного использования программы следует добавить удобный интерфейс, а именно значки действий.

Для этого добавляем файл ресурсов с расширением .qrc, помещаем изображения в формате .png размером 512x512 для каждого действия, далее через метод **setIcon**и класс **Qicon** связываем иконки с каждым соответственным действием. Результат изображен на рисунке 1.

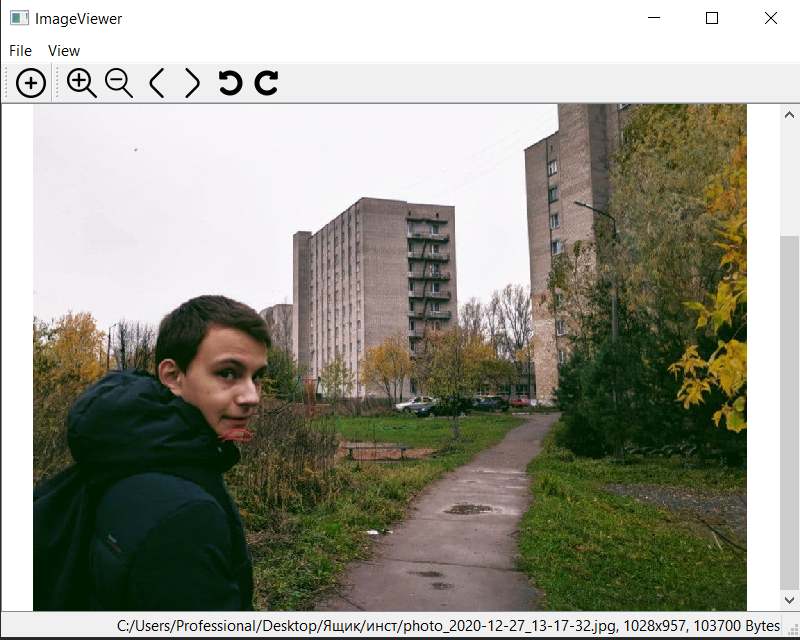


Рисунок 1- Значки

**Поворот и зум изображения**

Для зума(увеличение – уменьшения) изображения используется **QGraphicsView.scale** (рис. 2). Для поворота используются функции **rotateImageRight** и **rotateImageLeft** (рис. 3).

Примеры:

void MainWindow::**rotateImageRight**()

{

imageView->rotate(90);

}

void MainWindow::**rotateImageLeft**()

{

imageView->rotate(-90);

}

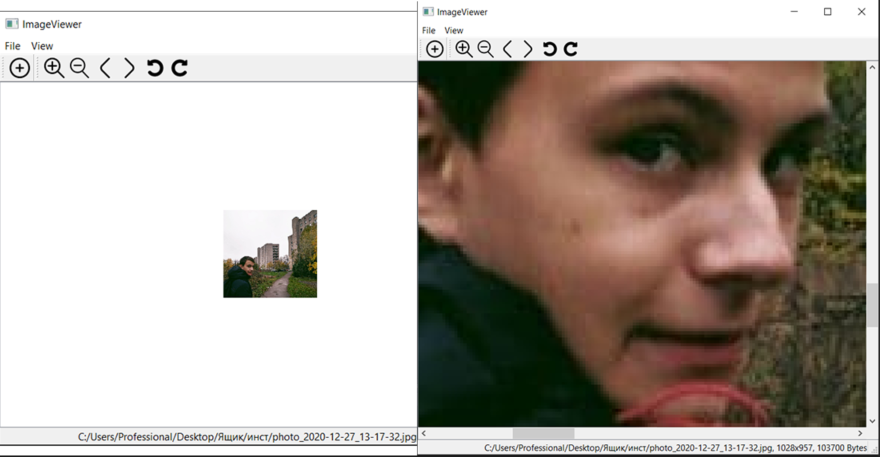


Рисунок 2 – Зум

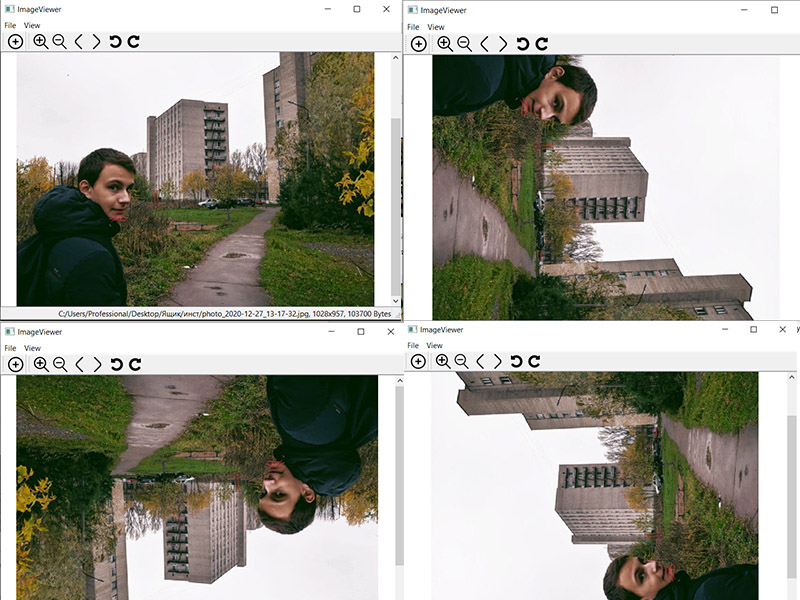


Рисунок 3 – Поворот изображения

**Moc, SIGNAL, SLOT**

**Метаобъектный компилятор** (сокр. МОК) (от англ.**Meta Object Compiler** или MOC) предназначен для генерации метаобъектного сопутствующего кода на основе библиотеки Qt. МОК считывает одно (или более) определение C++ класса из заголовочного файла языка C++ или исходного файла программы и генерирует один исходный файл на языке С++, содержащий метаобъектную информацию для классов. Получившийся исходный файл, сгенерированный МОК, должен быть скомпилирован и слинкован с реализацией класса (или может быть включен при помощи инструкции #include в исходный файл с описанием класса).

**Сигналы** (signals) - это методы, которые в состоянии осуществлять пересылку сообщений. Сигналы определяются в классе, как обычные методы, но без реализации. Они являются прототипами методов, содержащихся в заголовочном файле определения класса. Всю дальнейшую заботу о реализации кода для этих методов берет на себя препроцессор. Методы сигналов не должны возвращать каких-либо значений, поэтому перед именем метода всегда должно стоять void.

Сигнал не обязательно соединять со слотом. Если соединения не произошло, то он просто не будет обрабатываться.

**Слоты** (slots) — это методы, которые присоединяются к сигналам. По сути, они являются обычными методами. Основное их отличие состоит в возможности принимать сигналы. Как и обычные методы, они определяются в классе как *public*, *private* или *protected*. Соответственно, перед каждой группой слотов должно стоять одно из ключевых слов *private slots:*, *protected slots:* или *public slots:*

**Исходный код всех исходных файлов и файлов проекта**

Исходный код всех исходных файлов и файлов проекта доступен по ссылке: <https://github.com/borya178/Pattern_recognition_and_image_processing>

**Дистрибутив программы**

Дистрибутив был создан с помощью **windeployqt.exe.**

Дистрибутив был протестирован на виртуальной машине с ОС MS Windows 7, созданной с помощью программы VirtualBox.

Дистрибутив проекта в архиве доступен по ссылке: <https://github.com/borya178/Pattern_recognition_and_image_processing>  
Запустить программу можно, нажав на файл **Lab1.exe**.

**Вывод**

В ходе выполнения лабораторной работы я познакомился со средой Qt и разработал, и реализовал программу просмотра изображений.