МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Российский государственный университет им. А. Н. Косыгина   
(Технологии. Дизайн. Искусство)»

Кафедра информационных технологий

**Курсовая работа**

по дисциплине «Объектно-ориентированное программирование»

на тему: «Разработка приложения для определения номерных знаков автомобилей»

Выполнил:

студент 2 курса Сидоров Д. С.  
группа ИТС-123 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*подпись*

Проверил:   
к.т.н., доцент Семенов А.А.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*подпись*

Дата «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г.

РГУ им. А.Н. Косыгина, 2025 г.

Содержание

[**Введение** 3](#_Toc168256392)

[**Основная часть** 4](#_Toc168256393)

[Постановка задачи 4](#_Toc168256394)

[Обоснование выбора программных средств для решения задачи 4](#_Toc168256395)

[Описание использованных средств языка программирования 4](#_Toc168256396)

[Руководство пользователя 6](#_Toc168256397)

[**Листинг кода** 13](#_Toc168256398)

[main.cpp 13](#_Toc168256399)

[mainwindow.h 13](#_Toc168256400)

[mainwindow.cpp 15](#_Toc168256401)

[codeeditor.h 29](#_Toc168256402)

[codeeditor.cpp 31](#_Toc168256403)

[settingsdialog.h 39](#_Toc168256404)

[settingsdialog.cpp 40](#_Toc168256405)

[editorsettings.h 42](#_Toc168256406)

[editorsettings.cpp 42](#_Toc168256407)

[cppsyntaxhighlighter.h 43](#_Toc168256408)

[cppsyntaxhighlighter.cpp 44](#_Toc168256409)

[**Вывод** 47](#_Toc168256410)

[**Список литературы** 48](#_Toc168256411)

# Введение

В современном мире автоматизация процессов становится неотъемлемой частью нашей жизни. Одной из важных задач является автоматическое распознавание автомобильных номеров, которое находит применение в различных сферах: от управления парковками до систем безопасности и контроля дорожного движения.

Разработка приложения для определения номерных знаков автомобилей представляет собой комплексную задачу, требующую применения современных технологий компьютерного зрения и обработки изображений. Данная работа направлена на создание программного обеспечения, способного эффективно выделять и распознавать автомобильные номера на изображениях.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Разработка архитектуры приложения с использованием принципов ООП
2. Создание модуля захвата и предварительной обработки изображений
3. Реализация алгоритма обнаружения области номерного знака на изображении
4. Разработка системы распознавания символов на номерном знаке
5. Создание пользовательского интерфейса для взаимодействия с программой
6. Организация хранения и обработки полученных данных
7. Тестирование и отладка работы приложения

Для реализации проекта предлагается использовать следующие технологии и инструменты:

* Язык программирования: Python (версия 3.8 или выше)
* Библиотеки компьютерного зрения: OpenCV
* Библиотека для распознавания текста: Tesseract OCR
* Фреймворк для создания GUI: PyQt5
* Система управления базами данных: SQLite
* Среда разработки: PyCharm

Выбор Python обусловлен его простотой, большим количеством доступных библиотек для работы с изображениями и машинного обучения, а также широкими возможностями для создания графического интерфейса. OpenCV предоставляет необходимые инструменты для обработки изображений и выделения областей интереса, а Tesseract OCR является проверенным решением для распознавания текста.

Результатом выполнения курсовой работы должно стать функционирующее приложение с графическим интерфейсом, способное определять и распознавать автомобильные номера как на статических изображениях, так и в режиме реального времени при подключении камеры.

# Основная часть

## Постановка задачи

В данной работе необходимо создать редактор исходного кода для языка программирования C++, которая поддерживает автозаполнение и подсветку кода, работу с файлами, компиляцию и запуск программ. Среда должна быть простой в использовании, иметь простой в использовании интерфейс и обеспечивать все необходимые функции для комфортной работы программиста.

## Обоснование выбора программных средств для решения задачи

Для разработки среды разработки было решено использовать следующие инструменты и технологии:

* **Qt Framework**: мощный инструмент для разработки кроссплатформенных приложений с графическим интерфейсом. Qt предоставляет богатый набор библиотек для создания интерфейсов, работы с файлами, обработки событий и многого другого.
* **Qt Creator**: интегрированная среда разработки (IDE), разработанная для работы с Qt. Qt Creator предоставляет все необходимые инструменты для написания, редактирования, отладки и компиляции кода. В рамках данного проекта Qt Creator использовался для разработки интерфейса пользователя, написания кода на C++ и интеграции различных библиотек и фреймворков.
* **qmake**: это инструмент сборки, входящий в состав Qt. Он используется для автоматической генерации Makefile'ов, необходимых для сборки проектов. qmake позволяет описывать структуру проекта, зависимости, настройки компилятора и компоновщика в простом и понятном формате.
* **C++17**: основной язык программирования для реализации логики приложения, так как он обеспечивает высокую производительность и широкие возможности для работы с системными ресурсами.
* **GNU C++ (g++)**: компилятор для языка C++, который поддерживает все необходимые функции для компиляции и запуска приложений.

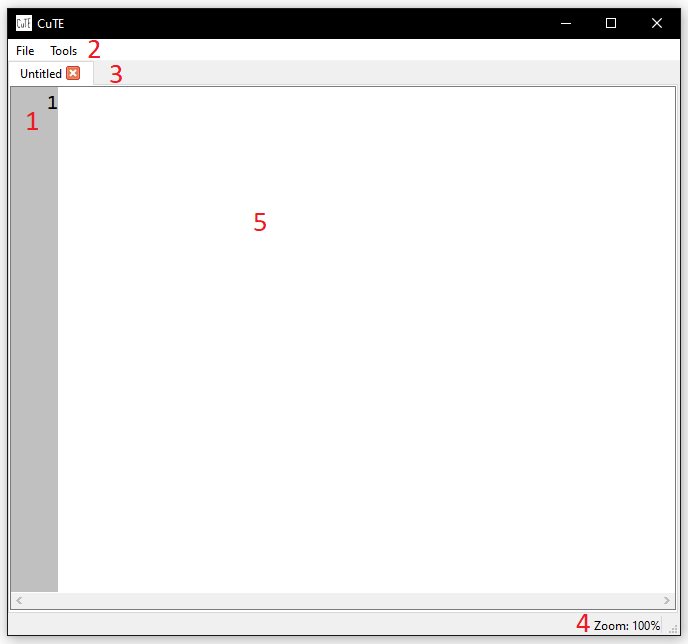
## Описание использованных средств языка программирования

* **Классы и объекты**: основная часть функционала приложения реализована с использованием классов и объектов. Были созданы классы для основной формы (MainWindow), формы настроек (settingsDialog), редактора кода (codeEditor) и других компонентов. Это позволило организовать код в логические блоки и улучшить его читаемость и поддерживаемость.
* **Наследование и полиморфизм**: для создания настраиваемых элементов интерфейса, таких как редактор кода с номерами строк и автозаполнением, использовалось наследование. Например, класс CodeEditor наследуется от QPlainTextEdit и расширяет его функциональность.
* **Указатели и динамическое распределение памяти**: для управления динамическими ресурсами, такими как диалоговые окна и процессы компиляции, использовались указатели. Динамическое распределение памяти позволило создавать и удалять объекты в процессе выполнения программы, что улучшило управление ресурсами.
* **Сигналы и слоты**: одна из ключевых особенностей Qt — механизм сигналов и слотов, который использовался для связи между компонентами. Сигналы и слоты позволяют передавать сообщения между объектами и обрабатывать события асинхронно. Это упрощает реализацию реактивного интерфейса пользователя.
* **Шаблоны**: в некоторых частях проекта использовались шаблоны, что позволило создавать обобщенные функции и классы, работающие с различными типами данных. Это улучшило гибкость и повторное использование кода.
* **Потоки**: для выполнения длительных операций, таких как компиляция и запуск программ, использовались потоки. Это позволило избежать блокировки пользовательского интерфейса и обеспечило более плавную работу приложения.
* **Работа с файлами**: для сохранения и загрузки исходного кода использовались стандартные возможности C++ для работы с файлами. В сочетании с возможностями Qt это позволило реализовать функции открытия, сохранения и автосохранения файлов.
* **Обработка исключений**: для улучшения надежности и устойчивости приложения использовались механизмы обработки исключений. Это позволило корректно обрабатывать ошибки, возникающие во время выполнения программы, и предоставлять пользователю информативные сообщения об ошибках.

Эти средства языка программирования C++ в сочетании с возможностями Qt позволили создать функциональное и удобное приложение для редактирования, компиляции и выполнения исходного кода на C++.

## Руководство пользователя

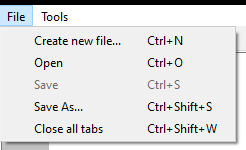
**Основное окно приложение (mainwindow, Рисунок 1)**

****

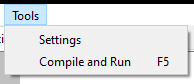
*Рисунок 1*

1. Поле для вывода номеров строки.

2. Поле виджетов для работы с файлами, открытия настроек, компиляции и выполнения кода, к которым можно обращаться с помощью горячих клавиш (Рисунки 2 и 3).



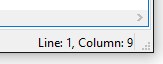
*Рисунок 2*



*Рисунок 3*

3. Поле для навигации между вкладками с открытыми файлами в них.

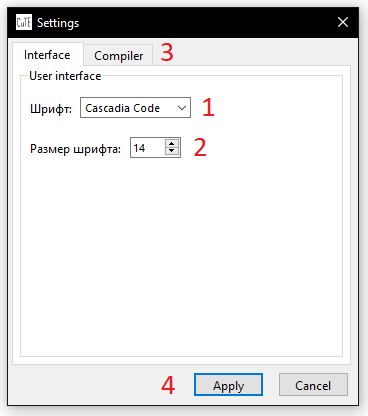
4. Статус-бар отображающий масштаб редактора, который можно изменять с помощью клавиш Ctrl + WheelUp/WheelDown, и положение активного курсора при редактировании текста (Рисунок 4).



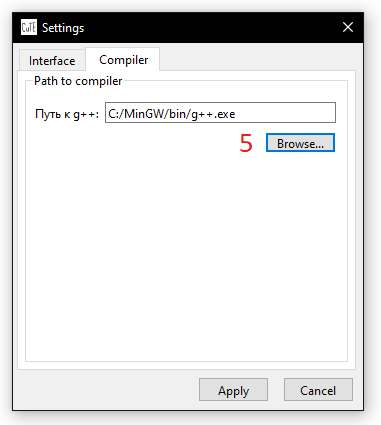
*Рисунок 4*

5. Редактор текста выбранного во вкладках файла.

**Окно настроек (settingsdialog, Рисунки 5 и 6)**

****

*Рисунок 5*

****

*Рисунок 6*

1. Поле для выбора шрифта в редакторе.

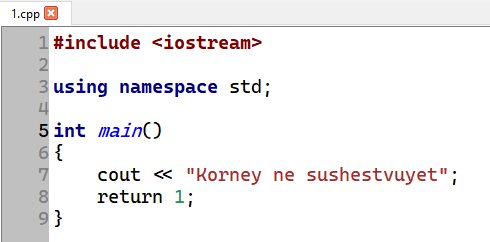
2. Поле для выбора размера шрифта.

3. Вкладки с настройками.

4. Кнопки для того, чтобы применить настройки, сохранить их и закрыть форму или закрыть форму без сохранения и применения.

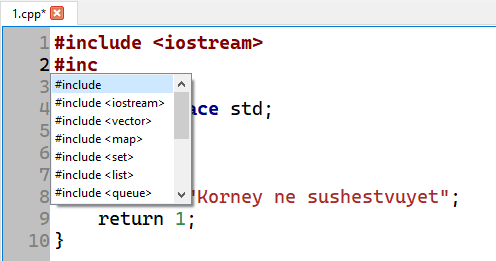
5. Кнопка для открытия диалогового окна с выбором пути к компилятору и сохранением его в соответствующее поле (в нашем случае это g++).

**Визуальное отображение функционала приложения**



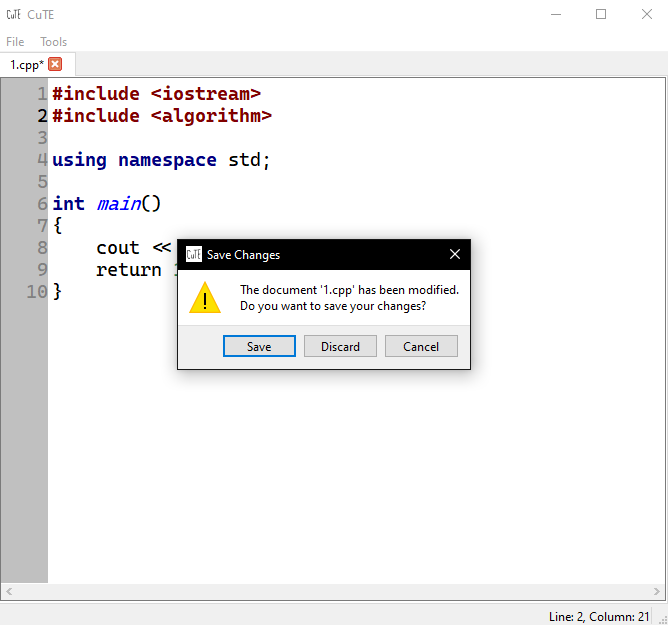
*Рисунок 7*

На Рисунке 7 можно увидеть реализацию функции подсветки синтаксиса кода на языке C++ и выделения активной строки.



*Рисунок 8*

На Рисунке 8 показана работа поля подсказок, автозаполнения по клавише Enter и отображения измененного файла знаком “\*” после его названия в поле с вкладками.

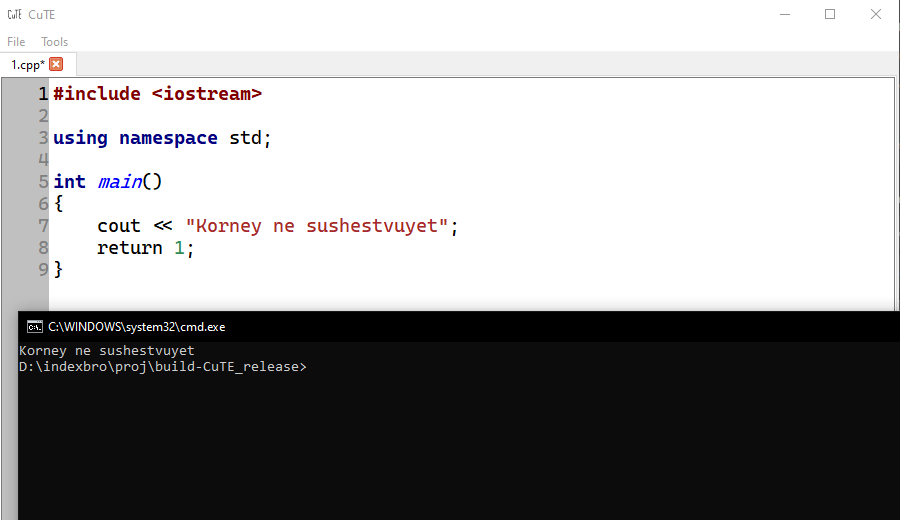


*Рисунок 9*

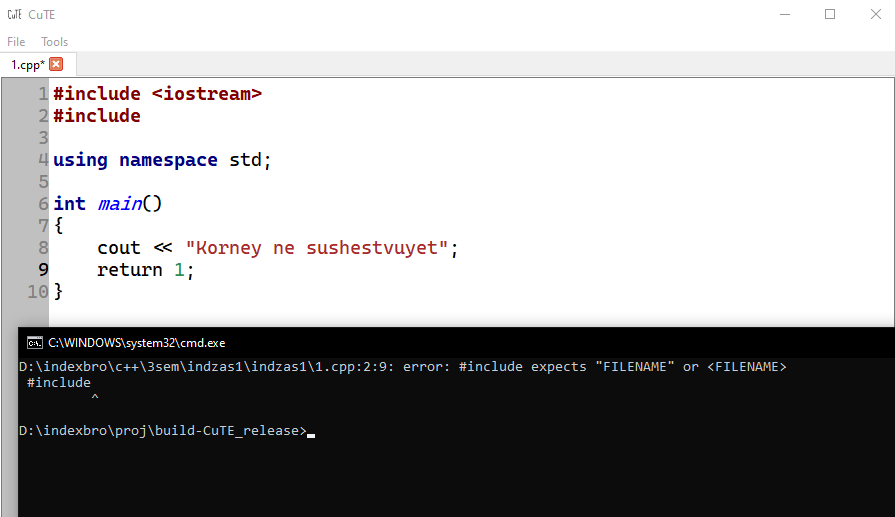
Рисунок 9 показывает действие программы на закрытие вкладки с файлом или завершение работы программы без предварительного его сохранения. Приложение предлагает применить сохранения, не учитывать их или отменить действие, которое привело к появлению этого окна.

Также стоит отметить систему табуляций, которую сложно отобразить визуально. Она работает автоматически, с учетом действующего отступа в редактируемом блоке кода, и в нее интегрировано проставление закрывающей фигурной скобки при открытии фрагмента.

Перед тем, как перейти к представлению функции компиляции и запуска исходного кода, стоит отметить, что этот модуль работает исключительно с компилятором g++ из коллекции GNU. С учетом того, что программа рассчитана на работу на платформе Windows, необходимо скачать набор инструментов MinGW, с которым поставляется нужный нам g++, а также проложить к нему путь в переменные среды системы (PATH), в частности указать директорию bin (например C:\MinGW\bin). После этого обязательным является задать путь до g++ в окне настроек программы, как показано на Рисунке 6.



*Рисунок 10*

**

*Рисунок 11*

По нажатию на кнопку Compile and Run или использованию горячей клавиши F5 происходит компиляция активного файла .сpp, создание исполняемого файла .exe в директории документа .cpp и его выполнение в новом открытом окне терминала (Рисунок 10).

В случае ошибок компиляции, они также отобразятся в терминале (Рисунок 11).

# Листинг кода

Проект состоит из 5 основных классов: mainwindow, codeeditor, settingsdialog, editorsettings и cppsyntaxhighlighter, а также файл main, содержащий основную функцию и точку входа для приложения.

## main.cpp

#include "mainwindow.h"

#include <QApplication>

#include <QStyleFactory>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

MainWindow w;

w.show();

return a.exec();

}

## mainwindow.h

*… программный код удален, так как это образец.*

# Вывод

В ходе выполнения курсовой работы на тему "Разработка приложения для редактирования, компиляции и выполнения исходного кода на C++" были достигнуты значимые результаты. Прежде всего, было создан редактор исходного кода, предоставляющее пользователям удобный и интуитивно понятный интерфейс для написания, компиляции и выполнения исходного кода на языке C++. Приложение поддерживает автозаполнение кода, подсветку синтаксиса, автоматические отступы и номера строк, что значительно упрощает процесс написания и отладки программного обеспечения.

Интеграция возможности компиляции и выполнения программ непосредственно из интерфейса приложения ускоряет цикл разработки и тестирования, позволяя пользователям компилировать и запускать программы одним нажатием кнопки. Реализация механизма сигналов и слотов Qt обеспечила создание реактивного и отзывчивого интерфейса, что улучшает пользовательский опыт.

Использование возможностей языка C++, таких как классы, объекты, наследование, полиморфизм и шаблоны, обеспечило модульность и расширяемость кода. Это позволяет легко поддерживать и расширять функциональность приложения в будущем. Механизмы автосохранения и обработки ошибок повышают надежность и устойчивость приложения, предупреждая пользователей о несохраненных изменениях перед закрытием приложения или файла.

Выбор программных средств, таких как Qt и Qt Creator, оказался обоснованным, так как они значительно упростили процесс разработки и предоставили мощные инструменты для создания пользовательских интерфейсов и управления проектами. Разработанное приложение успешно решает поставленные задачи и представляет собой функциональное средство для разработки программ на языке C++. Оно может быть использовано как учебный инструмент для студентов, так и как полезный инструмент для программистов, занимающихся разработкой на C++.

Исходный код проекта:

<https://github.com/folivorra/CuTE.git>

# Список литературы

1. Интернет-ресурс: https://www.learncpp.com/ (дата обращения 20.03.2024)

2. Интернет-ресурс: https://doc.qt.io/ (дата обращения 09.04.2024)

3. Интернет-ресурс: https://metanit.com/cpp/qt/ (дата обращения 15.03.2024)

4. Интернет-ресурс: https://github.com/ (дата обращения 12.05.2024)