Частное учреждение образование

«Колледжа бизнеса и права»

ПРОГРАММА ДЛЯ АНАЛИЗА ПРОГРАММНОГО КОДА НА ЯЗЫКЕ С++

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине

«Основы алгоритмизации и программирование»

КП Т.091020.401

Руководитель проекта (Шаляпин  Ю. В.)

Учащийся (Черняков   В. А.)

2022

Содержание

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3

КП Т.091020.401 ПЗ

Разраб.

Черняков В.А.

Провер.

Шаляпин Ю.В.

Реценз.

Н. Контр.

Утверд.

*Программа для анализа программного кода на языке С++*

Лит.

Листов

КБиП

38

У

[Введение 4](#_Toc106569211)

[1 Объектно-ориентированный анализ и проектирование записки для курсового проекта 5](#_Toc106569212)

[1.1 Сущность задачи 5](#_Toc106569213)

[1.2 Проектирование модели 6](#_Toc106569214)

[2 Вычислительная система 7](#_Toc106569215)

[2.1 Требование к аппаратным операционным ресурсам 7](#_Toc106569216)

[2.2 Инструменты разработки 7](#_Toc106569217)

[3 Проектирование задачи 8](#_Toc106569218)

[3.1 Требования к приложению 8](#_Toc106569219)

[3.2 Концептуальный прототип 8](#_Toc106569220)

[3.3 Организация данных 9](#_Toc106569221)

[3.4 Функции и элементы управления 9](#_Toc106569222)

[3.5 Проектирование справочной системы приложения 9](#_Toc106569223)

[4 Описание программного средства 10](#_Toc106569224)

[4.1 Общие сведения 10](#_Toc106569225)

[4.2 Входные и выходные данные 10](#_Toc106569226)

[5 Методика испытаний 11](#_Toc106569227)

[5.1 Технические требования 11](#_Toc106569228)

[5.2 Функциональное тестирование 11](#_Toc106569229)

[6 Применение 13](#_Toc106569230)

[6.1 Назначение программы 13](#_Toc106569231)

[6.2 Условия программы 13](#_Toc106569232)

[6.3 Справочная система 13](#_Toc106569233)

[Заключение 13](#_Toc106569234)

[Список информационных источников 16](#_Toc106569235)

[Приложение А. Текст программных модулей 17](#_Toc106569236)

[Приложение Б. Результат работы программы 37](#_Toc106569237)

[Приложение В. Блок-схемы работы алгоритмов 39](#_Toc106569238)

# Введение

Основной задачей является разработка на языке программирования С++ в среде разработки Microsoft Visual Studio программу для анализа программного кода на языке С++.

Данная задача является актуальна на сегодняшний день. Очень много программистов каждый день ведут работу с различными проектами.

В первом разделе «Объектно-ориентированный анализ и проектирование системы» раскрывается организационная сущность задачи, описывается предметная область и круг задач, которые должны быть автоматизированы. Описывается задача, перечисляются основные функции программы. Строится информационная модель, отражающая сущности задачи, их свойства и взаимосвязи.

Во втором разделе «Вычислительная система» содержится описание требований к аппаратным и операционным ресурсам, применяемым инструментам разработки.

Во третьем разделе «Проектирование задачи» содержится описание приложения, а именно: требования к графическому дизайну, описание доступа к данным, описание защиты информации от несанкционированного использования, описание механизма защиты, все накладные ограничения в процессе будущей эксплуатации на основании предметной области, текстовое и графическое описание пользовательского интерфейса, логическая и физическая структура данных, состав модулей и функций, разработанные и взаимосвязанные функции.

В четвёртом разделе «Описание программного средства» представлены общие сведения о программном средстве и его функциональном назначении, структуру входных и выходных данных.

В пятом разделе «Методика испытаний» описываются требования к техническим средствам для проведения испытаний, функциональное тестирование, требования к характеристикам программы применительно к условиям эксплуатации, требования к информационной и программной совместимости программного средства и области его применения.

В шестом разделе «Применение» приводится структура справочной системы, а также методика ее использования.

В заключении будет проанализировано созданное программное средства, определена степень соответствия поставленной задачи и выполненной работы.

# Объектно-ориентированный анализ и проектирование записки для курсового проекта

## Сущность задачи

Основной задачей является разработка на языке программирования С++ в среде разработки Microsoft Visual Studio программу для анализа программного кода на языке С++.

При разработке программ на различных языках программирования, программист должен иметь представление об текущем состоянии кодовой базы проекта. Данная проблема на сегодняшний день очень востребовано, поскольку очень много программистов не способны в должной степени качественно выполнить данную задачу в больших объёмах.

Задача анализа данных может быть решена использованием компьютерных процессов обработки данных. Программа способна быстро провести анализ данных и сообщить об состоянии кодовой базы, значительно превышая скорость программиста.

Программы анализа данных в основном применимы программистами в разработке проектов, позволяя программисту автоматизировать основные функции анализа данных. В перечень таких функций могут входить поиск функций (методов) или переменных.

На данный момент реализовано большое количество программ аналогов таких как Cppcheck, Vera++, RATS и другие. Для сравнения мы рассмотрим программу Cppcheck. Данная программа реализует статический анализ программного кода на языке программирования С++. Программа имеет открытый исходный код. Программа имеет мощную систему проверки программного кода на ошибки и возможные исключения, при этом отличается большим быстродействием.

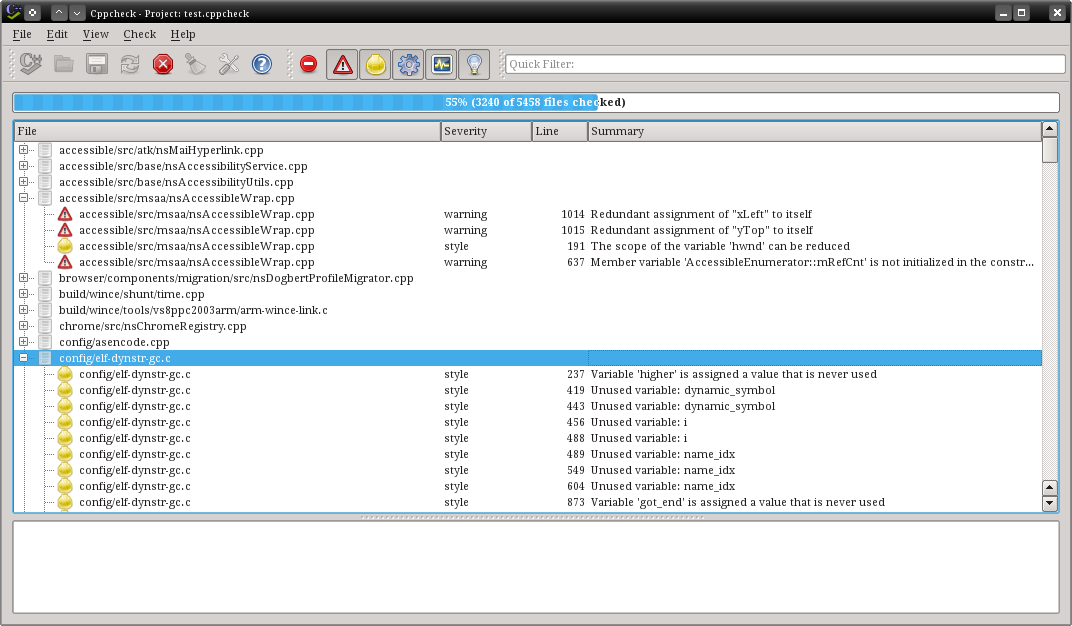


Рисунок 1.1 – Пример программы Сppcheck

Основным отличием разработанного ПС от аналог состоит в том, что разрабатываемый анализатор программного кода является более простым и менее требовательным в использовании. Разрабатываемая программа имеет удобный разработанный интерфейс, дружелюбную среду редактирования кода. Благодаря этому ПС не будет требовать огромных ресурсов операционный системы, но в тоже время реализовывать главный базовый функционал анализа программного кода, позволяющий получить основное представление об кодовой базе.

## Проектирование модели

Приложение представляет из себя начальную форму, в который пользователь может выполнить действие авторизации или регистрации. Во время выполнения этих действие присутствует определенная логика работы алгоритма, отраженная в приложении В.

После авторизации, пользователь попадает в основное окно, где он может по своему усмотрению выполнить действия, а именно: открыть файл, создать файл, сохранить файл, провести анализ программного кода, сменить локализацию, цветовое оформление, шрифт, получить справку, выйти в меню регистрации, выйти из приложения. Пользователь может комбинировать эти действия, получаю функциональное приложение, способное удовлетворить все потребности пользователя.

Всё приложение основано на принципах ООП. Каждый элемент имеет логику действия, закреплённую в классе. Все классы имеют определенные взаимосвязи, отраженные в диаграмме классов в приложении В.

# Вычислительная система

## Требование к аппаратным операционным ресурсам

Основные аппаратные характеристики ПК представляют из себя:

– один процессор частоты работы от 2 ГГц, архитектуры x64;

– оперативная память от 1 ГиБ;

– место на диске от 512 МиБ;

– клавиатура проводная;

– мышь проводная;

– видеопамять от 64 МиБ.

## Инструменты разработки

Для разработки была задействована ОС Microsoft Windows 10 Pro. Языки программирования С++, С++/CLI, HTML. Инструменты и среды разработки Microsoft Visual Code 2022, Windows Kits, HTML Help Workshop. Сетевое подключение не требуется.

Разработка производилась на ОС Microsoft Windows 10 Pro, поскольку именно на эту операционную систему ориентированно разрабатываемое ПС. В качестве языка программирования был выбран С++/CLI из-за хорошей совместимостью с Windows Form, для инструментов разработки была использована среда MS Visual Code 2022, имеющую поддержку разработки C++/CLI и Windows Form, так же было выбранное Windows Kits, позволяющий производить отладку приложения. HTML Help Workshop выбран из-за возможности лёгкого создания файла справки.

# Проектирование задачи

## Требования к приложению

Разрабатываемое ПС должно обеспечить организацию лаконичного и понятного пользователю интерфейса программы, средствами создания оконных форм с элементами меню для пользователя. Все кнопки должны быть отчётливо различимы и не иметь других смысловых нагрузок. Размещения элементов должно быть лаконичным, целесообразным и удобно сгруппированным.

Доступ к данным осуществляет через пользовательский интерфейс. Все данные располагаются в памяти программы.

В программе должен быть предусмотрена защита паролей пользователей от кражи. Рекомендуется использовать хеширование

Для корректной работы программы рекомендуется ограничить пользователя в регистрации. Пользователю запрещено регистрироваться под именем, под которым была уже произведена регистрация. Так же в приложении необходимо ограничить размеры вводимого логина и пароля от 4 до 20 символов.

Главным накладываемым ограничением является то, что бы пользователь мог без перебоев использовать приложение.

## Концептуальный прототип

Интерфейс пользователя будет представлять из себя окно, с областью для изменения содержимого файла и выводимой информации об результатах анализа программного кода в правой части окна. Пользователь сможет переключать вкладки результата анализа программного кода. Сверху должна располагаться строка меню, с выпадающими вкладками, позволяющее выбрать действие. Снизу должна располагаться строка состояния, на ней будет отображена информация о выбранной локализации приложения и индекс указателя. На рисунке 3.1 приведён концептуальный графический прототип интерфейса.

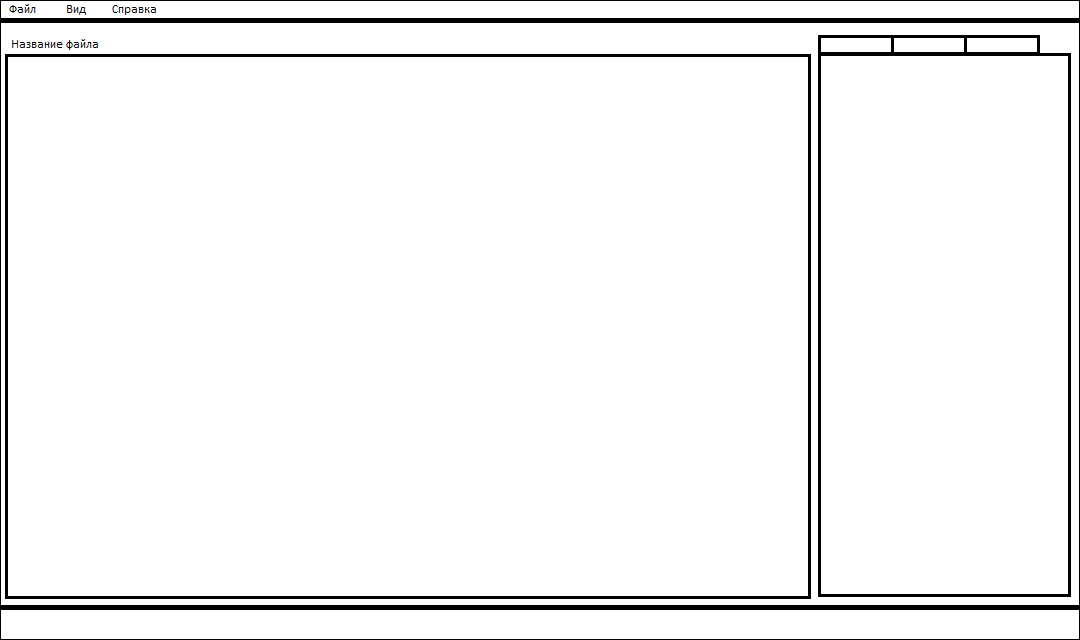


Рисунок 3. – Концептуальный графический прототип интерфейса

## Организация данных

Данные, логин и пароль пользователей располагаются в файле по указанному пути «%USERPROFILE%/users.cacpp». Остальные данные инициализируются во время запуска программы и располагаются в памяти программы и с завершением выполнения программы, данные уничтожаются.

Для работы с данными будет использован «Сборщик мусора» и базовые типы языка программирования C++/CLI. Это должно облегчить работу приложения при работе с данными и их высвобождением.

Всё программное решение должно быть расположено в пространстве имен CACPP, что позволит легко объединить функции в случаи использовании сторонних программных решение.

## Функции и элементы управления

В разрабатываемом ПС присутствую различные функции. Все функции приложения делятся на два типа: вызываемые внутренним обработчиком событий Windows Form и вызываемые функциями первого варианта. Из это следует сделать вывод, что вызов всех функций генерируется в обработчике событий Windows Form. К большинству событий привязаны функции, которые взаимодействую с друг другом через определённые параметры и флаги объекта MainWindow. Для правильной работы приложения обработчик событий вызывает функции приложения в одном потоке, что позволяет сохранить прослеживаемую последовательную логику работы приложения.

Большинство элементов интерфейса имеют ту или другую функции обработки определённого события. Каждая функция обработки событий принимает в себе два параметра, которые позволяют послать обработчику событий результат обработки и изменить поведения приложения.

Описание и текст всех функций приведён в приложение А.

Графическое описание классов и их функций представлено в приложении В.

## Проектирование справочной системы приложения

Справочная система реализована с помощью CHM-файла. Данный формат файла разработан компанией Microsoft и имеет встроенную поддержку в Windows Form. Файл справки представляет из себя каталог HTML-файлов в древовидной структуре. Всего в справке разрабатываемого приложения реализовано 3 страницы «Главная», «Анализатор кода», «Регистрация». Каждая страница имеет разъяснения по актуальным и более встречаемым вопросам. Общие вопросы размещены в «Главная». Вопросы по регистрации в «Регистрация». Вопросы по анализатору кода в «Анализатор кода».

Справочная системы была разработана в программе HTML Help Workshop. Данная программа создана компанией Microsoft, поэтому имеется поддержка со стороны системы. Программа представляет из себя редактор HTML страниц, которые позже можно объединить в CHM-файл. Кроме того, программа позволяет задать параметры поиска, организации дерева файлов и параметров просмотра.

Для вызова справочной системы используется элемент Help Provider, который является встроенным элементом Windows Form. При нажатии клавиши F1 в любом месте приложении, если не вызвано другого контекстного окна, вызывается справка приложения. Просмотр файла справки происходит с помощью встроенной программы просмотра, которая установлена на ОС Microsoft Windows 7 и более новых версиях.

# Описание программного средства

## Общие сведения

Приложение имеет название CACPP. Данное название образованно от сокращении слов названия «Code Analyzer CPP». Так же приложение имеет иконку в виде латинской буквы «V» жёлтого цвета.

Приложение имеет внешние модули, в которые вынесены ресурсы локализации, которые подключаются по мере подключения локализации в приложении.

Для функционирования программы необходимо иметь программное обеспечение Microsoft .Net Framework 4.8.

Процесс установки происходит путем запуска специального, заранее подготовленного, приложения. Пользователю нужно следовать инструкциям установщика для корректного завершения процесса установки. После установки на рабочем столе пользователя появится файл символической ссылки на исполняемый файл приложения.

Процесс удаления приложения заранее запланирован, при выполнении установки, и будет автоматически запущен при удалении приложения в настройках приложений Windows или «appwiz.cpl».

## Входные и выходные данные

В качестве входных данных программа принимает любой файл, который пользователь укажет в сплывающем окне. После этого программа откроет файл и его содержимое. Пользователь может редактировать содержимое файла.

Для вывода данных пользователь может сохранить свой файл. Так же после анализа кода будет представлен список всех методов, функций и переменных в правой части окна программы.

Дополнительными данными является файл паролей и логинов пользователей. Пользователь не имеет прямого достпука к управлению этим файлом.

# Методика испытаний

## Технические требования

Для испытаний разрабатываемого ПС требуется ПК со следующими характеристиками:

– один процессор частоты работы от 2 ГГц, архитектуры x64;

– оперативная память от 1 ГиБ;

– место на диске от 512 МиБ;

– клавиатура проводная;

– мышь проводная;

– видеопамять от 64 МиБ.

Так же, для проведения функционального тестирования были использованы:

– Microsoft Visual Studio 2022;

– Windows Kits.

## Функциональное тестирование

Функциональное тестирование — это тестирование ПО в целях проверки реализуемости функциональных требований, то есть способности ПО в определённых условиях решать задачи, нужные пользователям.

Функциональные требования определяют, что именно делает ПО, какие задачи оно решает. Функциональны требования включают в себя:

– Функциональная пригодность;

– Точность;

– Способность к взаимодействию;

– Соответствие стандартам и правилам;

– Защищённость.

Данное тестирование проводится для выявления неполадок и недочетов программы на этапе её сдачи в эксплуатацию.

Все исключающие ситуации и ошибки корректно обрабатываются в ПС и сообщаются пользователю в выпадающем окне. Пример представлен на рисунке 5.1.

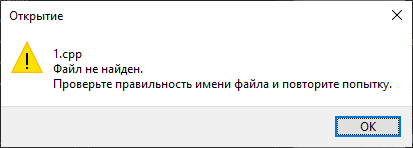


Рисунок 5.1 – Результат обработки исключения

Перед функциональным тестированием необходимо удостовериться в исправности работы базавой системы и удостовериться в том, что другие программы не перехватят события, вызванные обработчиком событий Windows Form.

Тестирование программы будет производиться последовательно, переходя из одной части программы в другую.

Результаты тестирования представлены в таблице 5.1. Фактические результаты работы представлены в приложении Б «Результат работы программы».

Таблица . – Тест-кейсы для функций

|  |  |  |  |
| --- | --- | --- | --- |
| № | Модуль/функция | Шаги воспроизведения | Результат |
| 1 | Запустить приложение CACPP | Открыть директорию приложения. Запустить двойным щелчком файл «cacpp.exe». | Ожидаемый: отображается формы регистрации. |
| Фактический: фактический результат представлен на рисунке Б.1 и фактический результат соответствует ожидаемому, все условия соблюдены. |
| 2 | Авторизоваться в приложении | Выбрать вкладку «Вход» на форме регистрации. Предварительно корректно заполнить поля «Логин» и «Пароль». Нажать кнопку «Подтвердить». | Ожидаемый: отображается основной формы приложения. |
| Фактический: фактический результат представлен на рисунке Б.2 и фактический результат соответствует ожидаемому, все условия соблюдены. |
| 3 | Открыть файл | Нажать вкладку меню «Файл». Выбрать в выпадающем списке пункт «Открыть файл…». В диалоговом окне выбрать файл. Нажать в диалоговом окне кнопку «Открыть». | Ожидаемый: отображение содержимого файла в области содержимого файла. |
| Фактический: фактический результат представлен на рисунке Б.3 и фактический результат соответствует ожидаемому, все условия соблюдены. Для примера был использован случайный файл. |
| 4 | Анализ программного кода | Нажать вкладку меню «Файл». Выбрать в выпадающем списке пункт «Анализ». | Ожидаемый: отображение результата анализа программного кода в правой части окна. |
| Фактический: фактический результат представлен на рисунке Б.4 и фактический результат соответствует ожидаемому, все условия соблюдены. |
| 5 | Сохранение файла | Нажать вкладку меню «Файл». Выбрать в выпадающем списке пункт «Сохранить файл…». | Ожидаемый: сохранения нового содержимого файла. |
| Фактический: фактический результат соответствует ожидаемому, все условия соблюдены. |

# Применение

## Назначение программы

Программа призвана помочь решить программисту задачу с формированием представлении об текущем состоянии кодовой базы проекта, путём анализа данных.

В приложении представлена защита данных пользователей, а именно хеширование пароля, для защиты его от кражи. Так же в программе реализуется принципы разграничения данных ООП. Каждый класс имеет приватные методы, доступ к которым производится только через заготовленные функции и методы.

В ходе программы были приняты исключения в разработке. Например, анализатор кода не проверяет код при каждом изменении, так как это потребовало бы больших аппаратных ресурсов. В связи с этим информация об анализе кода становится недоступной при изменении кода.

Еще одним исключение в разработке была решение об цветовом оформлении программы. Обычные средства Windows Form не позволяют изменить цветовое оформление элемента Tab Control. Поэтому было принято решение о создании цветового оформления, более сочетающегося со стандартными цветами Tab Control.

Так же в программе не рекомендуется открывать файлы расширения отличного от стандартных для языка программирования С++. Возможно некорректная работы программы. Стоит и учитывать размер файла, максимальный возможный размер файла, при котором сохраняется быстродействие программы пропорциональной зависит от аппаратных ресурсов.

Программа поддерживает кодировку UTF-16, но при использовании файлов с кодировкой, отличной от поддерживаемой, возможны некорректное отображение символов и их запись в сохранённый файл.

В программе не предусматривается сетевая поддержка из-за ненадобности в ходе эксплуатации программы.

## Условия применения

Для корректной работы приложения требуется соблюдать минимальные необходимые характеристики ПК, а именно:

– один процессор частоты работы от 2 ГГц, архитектуры x64;

– оперативная память от 1 ГиБ;

– место на диске от 512 МиБ;

– клавиатура проводная;

– мышь проводная;

– видеопамять от 64 МиБ;

– операционная система Microsoft Windows 10 Pro.

Для применения разрабатываемого ПС необходимо установить Microsoft .Net Framework 4.8. Данный компонент будет предложено установить в ходе установки программы.

## Справочная система

Справочная система реализована с помощью CHM-файла. Данный формат файла разработан компанией Microsoft и имеет встроенную поддержку в Windows Form. Файл справки представляет из себя каталог HTML-файлов в древовидной структуре. Всего в справке разрабатываемого приложения реализовано 3 страницы «Главная», «Анализатор кода», «Регистрация». Каждая страница имеет разъяснения по актуальным и более встречаемым вопросам. Общие вопросы размещены в «Главная». Вопросы по регистрации в «Регистрация”. Вопросы по анализатору кода в «Анализатор кода».

Справочная системы была разработана в программе HTML Help Workshop. Данная программа создана компанией Microsoft, поэтому имеется поддержка со стороны системы. Программа представляет из себя редактор HTML страниц, которые позже можно объединить в CHM-файл. Кроме того, программа позволяет задать параметры поиска, организации дерева файлов и параметров просмотра.

Для вызова справочной системы используется элемент Help Provider, который является встроенным элементом Windows Form. При нажатии клавиши F1 в любом месте приложении, если не вызвано другого контекстного окна, вызывается справка приложения. Просмотр файла справки происходит с помощью встроенной программы просмотра, которая установлена на ОС Microsoft Windows 7 и более новых версиях.

# Заключение

Были разработаны и созданы базовые компоненты программного обеспечения, реализующего программу для анализа программного кода языка программирования С++.

Для разработки ПС и достижению поставленной задачи были изучено и сформировано знания об предметной области «Основы алгоритмизации и программирования».

В ходе разработки ПС, были применены различные программные и аппаратные решения. Были изучены основы Microsoft .Net Framework 4.8 в связке с Windows Form и Windows API. Были изучены основы синтаксиса С++ и все исключающие возможности языка, для корректной работы анализатора программного кода. Так же были применены весь ранее изученный матерьял данной дисциплины. Основана технология создания справки средствами Windows Form и CHM-файлов.

За все время разработки было разработанное достаточное количество различных классов, что способствовало развитию и закреплению основ и принципов ООП. Так же были закреплены основы работы в среде разработке Microsoft Visual Studio 2022.

Сформированы и закреплены знания работы с созданием и управлением установщика программ в среде ОС Mircosoft Windows 10 Pro.

В процессе разработки программы использовался в большом объеме материал по программированию, что способствовало закреплению наработанных навыков и умений в этих областях знаний.

В программе реализованы такие задачи как: авторизация в системе как пользователь или администратор, удаление, изменение или создание файлов, анализ программного кода языка программирования С++, вывод информации об результатах анализа кода в список или подсветка синтаксиса текста в области с содержимым файла, для удобного просмотра. Так же было реализованы такие задачи, как: изменения локализации программы, изменение цветового оформления заднего фона, изменения шрифта текста области с содержимом файла, удобная справочная система.

Описание всех разработанных функций было рассмотрено в четвёртом разделе «Описание программного средства».

Преимуществом данного ПС являются такие свойства как: относительно лёгкое к установке программа, маленький вес программы, доступность локализации, простота функционирования, устройство-независимость.

Недостатками данного ПС является такие свойства как: ограниченный функционал, отсутствие кроссплатформенности.

Разработка и сопровождение данного разработанного ПС реализовано должным образом.

Разработанное ПС полностью отлажена и протестирована.

Поставленные задачи выполнены.

Для обеспечения надежности функционирования программы была проведена опытная эксплуатация.

Разработанное ПС является готов продуктом, готовым к использованию.

# Список информационных источников

1. Багласова, Т.Г. Методические указания по оформлению курсовых и дипломных проектов / Т.Г. Багласова, К.О. Якимович. – Минск : КБП , 2013 . – 29 c.
2. Багласова, Т.Г. Методические указания по оформлению курсовых работ, дипломных проектов и отчётов для учащихся специальности 2-40 01 01 «ПОИТ» / Т.Г. Багласова. – Минск : КБП , 2016 . – 30 с.
3. Дейтел, Х. Как программировать на C++ / Х. Дейтел, П. Дейтел. – М. : ДМК Пресс , 2018 . – 1021 с.
4. Каталог API (Microsoft) материалов [Электронный ресурс]. – Microsoft , 2022 – Режим доступа : https://docs.microsoft.com/en-us/windows/win32/appindex/windows-api-list. – Дата доступа : 24.05.2022.
5. Мейерс, С. Эффективное использование С++. 55 верных способов улучшить структуру и код ваших программ / С. Мейерс. – М. : ДМК Пресс , 2006 . – 300 с.
6. Объектно-ориентированный анализ и проектирование с примерами приложений / Гради Буч [и др.]. – 3-е изд. - М. : ООО «И.Д. Вильямс» , 2008 . – 720 с.
7. Общие требования к тестовым документам : ГОСТ 2.105-95. – Введ. 01.01.1996 . – Минск : Межгос. совет по стандартизации, метрологии и сертификации , 1995 . – 84 с.
8. Описание программы. Требования к содержанию, оформлению и контролю качества : ГОСТ 19.402-2000 . – Введ. 01.09.2001 . – Минск : Межгос. совет по стандартизации, метрологии и сертификации , 2000 . – 14 с.
9. Текст программы. Требования к содержанию, оформлению и контролю качества : ГОСТ 19.401-2000 . – Введ. 01.09.2001 . – Минск : Межгос. совет по стандартизации, метрологии и сертификации , 2000 . – 16 с.

# Приложение А (обязательное) Текст программных модулей

#include "RegistrationWindow.h"

#include <Windows.h>

using namespace CACPP;//основное пространства имён программы

[STAThreadAttribute] //подключаем атрибут для корректной работы программы в потоке

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, INT) {//главная инициализирующая функция

Application::EnableVisualStyles();//задаем видимый стиль

Application::SetCompatibleTextRenderingDefault(false);//отключает стандартный рендеринг текста

Application::Run(gcnew RegistrationWindow);//запускаем форму регистрации

return 0;

}

#pragma once

#include "Registration.h"

#include "MainWindow.h"

namespace CACPP {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

public ref class RegistrationWindow : public System::Windows::Forms::Form

{

public: void show(bool logout) {//функция делегат отвечающая за закрытие окна

logout ? this->Show() : this->Close();// в зависимости от параметра показывает или закрывает окно

}

public: void local(int l) {//функция делегат устанавливающая локализацию

this->comboBox1->SelectedIndex = l;//устанавливает правильную строку в выпадающем списке

this->selected(nullptr, nullptr);//меняет локализацию

}

private: System::Void button1Click(System::Object^ sender, System::EventArgs^ e) {//функция обработки нажатия кнопки "Подтвердить" во вкладке "Вход".

if ((this->textBox1->Text->Length <= MAX\_SIZE\_REG && this->textBox1->Text->Length > 3) ||

(this->textBox2->Text->Length <= MAX\_SIZE\_REG && this->textBox2->Text->Length > 3)) {//проверка длины логина и пароля

Registration^ r = gcnew Registration();//создания класса регистрации

bool admin = false;//преждевременная установка флага администратора.

if (r->login(this->textBox2->Text, this->textBox1->Text, 1, this->checkBox1->Checked)) {//попытка входа от имени администратора

admin = true;//устанавливаем флаг в случаи успеха

}

else {

if (!r->login(this->textBox2->Text, this->textBox1->Text, 0, this->checkBox1->Checked)) {//попытка входа от имени пользователя

MessageBox::Show(this->TL1->Text);//вывод сообщении об провале

return;

}

}

r->save();//сохраняем состояния входа

this->Hide();//прячем текущую форму

MainWindow^ mainWindow = gcnew MainWindow(admin,//создаем основное окно с передачей параметров

gcnew ReturnFun(this, &RegistrationWindow::show),//передаем делегат открытия окна

gcnew LocalFun(this, &RegistrationWindow::local));//передаем делегат локали

mainWindow->Show();//отображаем основное окно

}

else {

MessageBox::Show(this->TL2->Text);//выводим сообщение об провале

}

}

//функция обработки нажатия кнопки "Подтвердить" во вкладке "Регистрация".

private: System::Void button2Click(System::Object^ sender, System::EventArgs^ e) {

bool regok = false;//флаг успешной регистрации

if ((this->textBox3->Text->Length <= 20 && this->textBox3->Text->Length > 3) ||

(this->textBox4->Text->Length <= 20 && this->textBox4->Text->Length > 3) ||

(this->textBox5->Text->Length <= 20 && this->textBox5->Text->Length > 3)) {//проверка длины логина и пароля

Registration^ r = gcnew Registration();//создания класса регистрации

if (this->textBox5->Text == " ") {//проверка на регистрация администратора

if (r->addUser(this->textBox3->Text, this->textBox4->Text, 1)) {//регистрируем администратора

MessageBox::Show(this->TL3->Text);//выводим сообщение об успехе

regok = true;//устанавливаем флаг успешной регистрации

}

else {

MessageBox::Show(this->TL4->Text);//выводим сообщение об провале

}

}

else {

if (this->textBox4->Text == this->textBox5->Text) {//сравниваем пароль и подтверждение пароля

if (r->addUser(this->textBox3->Text, this->textBox4->Text, 0)) {//регистрируем пользователя

MessageBox::Show(this->TL3->Text);//выводим сообщение об успехе

regok = true;//устанавливаем флаг успешной регистрации

}

else {

MessageBox::Show(this->TL4->Text);//выводим сообщение об провале

}

}

else {

MessageBox::Show(this->TL5->Text);//выводим сообщение об провале

}

}

r->save();//сохраняем состояния входа

}

else {

MessageBox::Show(this->TL1->Text);//выводим сообщение об провале

}

if (regok) {//если регистрация успешна, то очищаем содержимое текстбоксов и переходим на вкладку "Вход"

this->textBox3->Text = "";

this->textBox4->Text = "";

this->textBox5->Text = "";

this->tabControl1->SelectTab(0);

}

}

private: System::Void regwinLoad(System::Object^ sender, System::EventArgs^ e) {//функция загрузки окна

this->helpProvider1->SetShowHelp(this,true);//устанавливаем справку

Registration^ r = gcnew Registration();//создания класса регистрации

if (r->isLogin()) {//если есть пользователь в системе, то подставляем его логин и пароль.

this->textBox2->Text = r->getLogin();

this->textBox1->Text = r->getPassword();

this->checkBox1->Checked = true;

}

}

private: System::Void selected(System::Object^ sender, System::EventArgs^ e) {//функция смены локализации

//меняем локализацию в соответствии с выбранным элементом.

switch (this->comboBox1->SelectedIndex)

{

case 0: {

System::Threading::Thread::CurrentThread->CurrentUICulture = gcnew System::Globalization::CultureInfo("en");

break;

}

case 1: {

System::Threading::Thread::CurrentThread->CurrentUICulture = gcnew System::Globalization::CultureInfo("ru");

break;

}

case 2: {

System::Threading::Thread::CurrentThread->CurrentUICulture = gcnew System::Globalization::CultureInfo("be");

break;

}

}

//сохраняем значения, ведь при очисти они пропадут

String^ byf = this->textBox1->Text;

String^ byf2 = this->textBox2->Text;

String^ byf3 = this->textBox3->Text;

String^ byf4 = this->textBox4->Text;

String^ byf5 = this->textBox5->Text;

bool byf6 = this->checkBox1->Checked;

int buf7 = this->tabControl1->SelectedIndex;

this->Controls->Clear(); //очищаем элементы

InitializeComponent();//заново инициализируем элементы с нужной локализацией

this->textBox1->Text = byf;

this->textBox2->Text = byf2;

this->textBox3->Text = byf3;

this->textBox4->Text = byf4;

this->textBox5->Text = byf5;

this->checkBox1->Checked = byf6;

this->tabControl1->SelectedIndex = buf7;

}

};

}

#include "Registration.h"

namespace CACPP {

Registration::Registration() {

this->path = System::Environment::GetEnvironmentVariable("USERPROFILE")+"\\users.cacpp";//определяем путь к файлу с паролями

this->users = gcnew List<User^>();//инициализируем список пользователей

this->nameL = gcnew String("");//инициализируем логин

this->passL = gcnew String("");//инициализируем пароль

try {

StreamReader^ sr = gcnew StreamReader(this->path);//создаем поток чтения файла

String^ line = sr->ReadLine();//читаем первую строку

if (line == "true") {//проверяем, если пользователь в системе

this->logF = true;//устанавливаем флаг входа

this->nameL = sr->ReadLine();//устанавливаем логин

this->passL = sr->ReadLine();//устанавливаем пароль

}

line = sr->ReadLine(); //читаем следующую строку

while (line != nullptr) {//продолжаем цикл, пока строка не пуста

User^ user = gcnew User();//инициализируем пользователя

user->login = line;//задаем логин

line = sr->ReadLine();//читаем строку

user->hash = line;//задаем пароль

line = sr->ReadLine();//читаем строку

this->users->Add(user);//добавляем пользователя в список

}

sr->Close();//закрываем поток

}

catch (FileNotFoundException^ ) {

//исключение, если файл не обнаружен

//данное исключение не нарушает работу программы и должно игнорироваться

}

catch (UnauthorizedAccessException^ ) {

//исключение на доступ к файлу

//в случаи исключения вызываем предупреждение

System::Windows::Forms::MessageBox::Show("Access denied, you must have administrator rights.");

}

}

void Registration::save() {//сохранение пользователей в файл

try {

StreamWriter^ sw = gcnew StreamWriter(this->path);//создаем поток записи файла

String^ a = (this->logF) ? "true" : "false"; //определяем первую строку файла

sw->WriteLine(a);//записывает первую строку файла

if (logF) {//если флаг вход истинный, то сохраняем пользователя

sw->WriteLine(this->nameL);//записываем логин

sw->WriteLine(this->passL);//записывает пароль

}

for each (User ^ u in this->users) {//проходимся по всему списков пользователей

sw->WriteLine(u->login);//записываем логим

sw->WriteLine(u->hash);//записывае пароль

}

sw->Close();//закрываем поток

}

catch (UnauthorizedAccessException^ ) {

//исключение на доступ к файлу

//в случаи исключения вызываем предупреждение

System::Windows::Forms::MessageBox::Show("Access denied, you must have administrator rights.");

}

}

bool Registration::login(String^ name, String^ password, bool admin, bool loggedin) {//вход пользователя в систему

for each (User ^ u in this->users) {//перебираем пользователей в списке

if (u->login == name) {//если логин совпадает производим вход

String^ a = (gcnew String(password->Copy(password) + ((admin) ? ("+") : ("-"))))->GetHashCode().ToString();//определяем параметры входа администратора и хешируем

if (u->hash == a) {//сравниваем пароли

if (loggedin) {//если пользователь хочет остаться в системе

this->logF = true;//устанавливаем флаг входа в систему

this->nameL = name;//сохраняем логин

this->passL = password;//сохраняем пароль

}

else {

this->logF = false;//устанавливаем флаг входа в систему

}

return true;//возвращаем успешный результат входа

}

}

}

return false;//возвращаем отрицательный результат входа

}

bool Registration::addUser(String^ name, String^ password, bool admin) {//регистрация пользователя

for each (User ^ u in this->users) {//перебираем пользователей и ищем совпадению по логину

if (u->login == name) {

return false;//если логин совпадает, отменяем регистрацию

}

}

User^ user = gcnew User();//создаем пользователя

user->login = name;//задаем логин

user->hash = (gcnew String(password->Copy(password) + ((admin) ? ("+") : ("-"))))->GetHashCode().ToString();//задаем пароль и хешируем

this->users->Add(user);//добавляем пользователя в список

return true;//возвращаем успешный результат регистрации

}

bool Registration::isLogin() {//возвращает параметры входа

return this->logF;

}

String^ Registration::getPassword() {//возвращает пароль

return this->passL;

}

String^ Registration::getLogin() {//возвращает логин

return this->nameL;

}

}

#pragma once

#include "FileCPP.h"

#include "Analyzer.h"

#ifdef \_DEBUG

#define debug(x) if(this->admin){MessageBox::Show(x);}

#else

#define debug(x)

#endif

namespace CACPP {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

public delegate void ReturnFun(bool);//делегат открытия окна

public delegate void LocalFun(int l);//делегат локали

public ref class MainWindow : public System::Windows::Forms::Form

{

ref struct GenSize { //Структура для сохранения начальной позиции элементов

int WinH; //окна

int WinW;

int TextH; //тексбокса

int TextW;

int PanelH; //боковой панели

int PanelWPoint;

int PanelListH;

};

public:

MainWindow(bool admin, ReturnFun^ rf, LocalFun^ lf) {

InitializeComponent();

this->admin = admin;//сохраняет статус администратора

this->lf = lf;//ссылка на делегат для локали

this->rf = rf;//ссылка на делегат для открытия окна

this->filecpp = gcnew FileCPP("New");//инициализация файла

this->label1->Text = filecpp->Path + ((this->filecpp->Status) ? "\*" : "");//устанавливаем название

//this->regWin = regWin;

}

//Функции нажатия кнопки на смену локализации

private: System::Void menuLanguageRu(System::Object^ sender, System::EventArgs^ e) {//русский

this->lf(1);

update("ru");

}

private: System::Void menuLanguageEn(System::Object^ sender, System::EventArgs^ e) {//английский

this->lf(0);

update("en");

}

private: System::Void menuLanguageBe(System::Object^ sender, System::EventArgs^ e) {//беларуский

this->lf(2);

update("be");

}

private: void update(String^ language) {//функция смены локализации

this->textChangedEnable = false;//отключаем функцию изменения текста

System::Threading::Thread::CurrentThread->CurrentUICulture = gcnew System::Globalization::CultureInfo(language);//изменяем локализацию приложения

//сохраняем значения, ведь при очисти они пропадут

String^ byf = this->richTextBox1->Text;

String^ byf2 = this->label1->Text;

String^ byf3 = this->listBox1->Text;

String^ byf4 = this->listBox2->Text;

String^ byf5 = this->listBox3->Text;

int buf6 = this->Height;

int buf7 = this->Width;

this->Controls->Clear();//очищаем элементы

InitializeComponent();//заново инициализируем элементы с нужной локализацией

//восстанавливаем размер окна

this->Height = buf6;

this->Width = buf7;

winResize(nullptr,nullptr);//обновляем размер элементов

colorUpdate();//обновляем цвета

//восстанавливаем остальные значения

this->richTextBox1->Text = byf;

this->label1->Text = byf2;

this->listBox1->Text = byf3;

this->listBox2->Text = byf4;

this->listBox3->Text = byf5;

this->textChangedEnable = true;//включаем функцию изменения текста

if (!this->admin) {//прячем элементы администратора

this->debugToolStripMenuItem->Enabled = false;

this->debugToolStripMenuItem->Visible = false;

}

}

private: System::Void winLoad(System::Object^ sender, System::EventArgs^ e) {//функция загрузки окна

this->FormClosing += gcnew System::Windows::Forms::FormClosingEventHandler(this, &MainWindow::winClosing);//добавляем функция закрытия окна

if (!this->admin) {//прячем элементы администратора

this->debugToolStripMenuItem->Enabled = false;

this->debugToolStripMenuItem->Visible = false;

}

//сохраняем изначальный размер для последующего его изменения

this->size = gcnew GenSize();

this->size->WinH = this->Height;

this->size->WinW = this->Width;

this->size->TextH = this->richTextBox1->Height;

this->size->TextW = this->richTextBox1->Width;

this->size->PanelH = this->tabControl1->Height;

this->size->PanelWPoint = this->tabControl1->Location.X;

this->size->PanelListH = this->listBox1->Height;

menuBackgroundWhite(nullptr, nullptr);//красим по умолчанию в белый фон

//красим и отключаем листбоксы

this->listBox1->ForeColor = System::Drawing::SystemColors::ScrollBar;

this->listBox2->ForeColor = System::Drawing::SystemColors::ScrollBar;

this->listBox3->ForeColor = System::Drawing::SystemColors::ScrollBar;

this->listBox1->Enabled = false;

this->listBox2->Enabled = false;

this->listBox3->Enabled = false;

}

private: System::Void winClosing(System::Object^ sender, System::Windows::Forms::FormClosingEventArgs^ e) {//функция перед закрытием окна

e->Cancel = !open();//вызываем функцию для досрочного сохранения файла. Результат будет применен к остановке закрытия окна или нет.

}

private: bool clo = false;//параметр указывающий на закрытие окна или выход в меню регистрации

private: System::Void winClosed(System::Object^ sender, System::Windows::Forms::FormClosedEventArgs^ e) {

this->rf(this->clo);//вызов делегата меню регистрации, который закроет окно как необходимо

}

//функция выхода из системы

private: System::Void menuLogout(System::Object^ sender, System::EventArgs^ e) {

this->clo = true;

this->Close();

}

private: bool textChangedEnable = true;//параметр указывающий на работу функции изменения текста

private: System::Void textChanged(System::Object^ sender, System::EventArgs^ e) {

if (this->textChangedEnable) {

if (this->filecpp->Path == "New" && this->richTextBox1->Text == "") {//проверка на пустой новый файл

this->filecpp->Status = false;//устанавливаем статус

this->label1->Text = filecpp->Path + ((this->filecpp->Status) ? "\*" : "");//устанавливаем название файла

}

else {

this->filecpp->Status = true;//устанавливаем статус

this->label1->Text = filecpp->Path + ((this->filecpp->Status) ? "\*" : "");//устанавливаем название файла

}

if (this->colorOn) {//отключаем постоянный сброс цвета, что бы окно не мерцало

int buf = this->richTextBox1->SelectionStart;//сохраняем указатель в тексте

this->richTextBox1->SelectAll();//выделяем все

this->richTextBox1->SelectionColor = System::Drawing::SystemColors::WindowText;//красим

this->richTextBox1->Select(buf, 0);//восстанавливаем указатель

this->colorOn = false;//запрещаем сброс цвета

}

//красим и отключаем листбоксы

this->listBox1->ForeColor = System::Drawing::SystemColors::ScrollBar;

this->listBox2->ForeColor = System::Drawing::SystemColors::ScrollBar;

this->listBox3->ForeColor = System::Drawing::SystemColors::ScrollBar;

this->listBox1->Enabled = false;

this->listBox2->Enabled = false;

this->listBox3->Enabled = false;

}

}

private: void save(bool as) {//функция сохранения

if (as) {//спросить путь

this->saveFileDialog1->FileName = filecpp->Path;//устанавливаем заготовленный путь

if ((this->saveFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::Cancel)) {

return;//отменяем если нет

}

this->filecpp->Path = this->saveFileDialog1->FileName;// получаем новый путь

}

if (this->filecpp->Path == "New") {//если файл новый

save(true);//спрашиваем путь в функции сохранения еще раз

return;

}

this->filecpp->Text = this->richTextBox1->Text;//передаем содержимое текста в файл

this->filecpp->save();//сохраняем файл

this->filecpp->Status = false;//устанавливаем стату

this->label1->Text = filecpp->Path + ((this->filecpp->Status) ? "\*" : "");//устанавливаем название файла

}

private: bool open() {//функция проверки открытия файла

if (this->filecpp->Status) {//проверяем если файл изменен

System::Windows::Forms::DialogResult^ dr;//Результат всплывающего окна

dr = MessageBox::Show(this->TL2->Text, this->TL1->Text, MessageBoxButtons::YesNoCancel);//вызываем окно

if (dr->Equals(System::Windows::Forms::DialogResult::Yes)) {//кнопка да

save(false);//сохраняем файл

return true;//посылаем продолжение операции

}

else if (dr->Equals(System::Windows::Forms::DialogResult::No)) {//кнопка нет

return true;//посылаем продолжение операции

}

else if (dr->Equals(System::Windows::Forms::DialogResult::Cancel)) {//кнопка отмены

return false;//посылаем отмену операции

}

}

return true;//посылаем продолжение операции

}

private: System::Void openFile(System::Object^ sender, System::EventArgs^ e) {//открытие файла

if (open()) {//сохраняем прошлый файл, если он не сохранен и проверяем продолжить ли действие

if ((this->openFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::Cancel)) {//вызываем диалоговое окно с выбором файла

return;//отменяем если нет

}

this->filecpp = gcnew FileCPP(this->openFileDialog1->FileName);//инициализируем новый файл

this->richTextBox1->Text = this->filecpp->Text;//передаем содержимое файла в текстбокс

this->filecpp->Status = false;//устанавливаем стату

this->label1->Text = filecpp->Path + ((this->filecpp->Status) ? "\*" : "");//устанавливаем название файла

}

}

private: System::Void newFile(System::Object^ sender, System::EventArgs^ e) {//функция нового файла

if (open()) {//сохраняем прошлый файл, если он не сохранен и проверяем продолжить ли действие

this->richTextBox1->Text = "";//обнуляем содержимое текстбокса

this->filecpp = gcnew FileCPP("New");//инициализируем новый файл

this->label1->Text = filecpp->Path + ((this->filecpp->Status) ? "\*" : "");//устанавливаем название файла

}

}

private: System::Void saveFileAs(System::Object^ sender, System::EventArgs^ e) {//функция обработки нажатия кнопки "Сохранить файл как..."

save(true);

}

private: System::Void saveFile(System::Object^ sender, System::EventArgs^ e) {//функция обработки нажатия кнопки "Сохранить как..."

save(false);

}

private: System::Void menuBackgroundWhite(System::Object^ sender, System::EventArgs^ e) {//функция обработки нажатия кнопки "Белый"

//инициализируем цвета под цветовую палитру

this->color1 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));//b

this->color2 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(240)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(240)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(240)));//b

this->color3 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));//b

this->color4 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));//b

colorUpdate();

}

private: System::Void menuBackgroundPink(System::Object^ sender, System::EventArgs^ e) {//функция обработки нажатия кнопки "Розовый"

//инициализируем цвета под цветовую палитру

this->color1 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(192)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(192)));//b

this->color2 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(170)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(170)));//b

this->color3 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(210)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(210)));//b

this->color4 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(200)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(200)));//b

colorUpdate();

}

private: System::Void menuBackgroundSkyblue(System::Object^ sender, System::EventArgs^ e) {//функция обработки нажатия кнопки "Голубой"

//инициализируем цвета под цветовую палитру

this->color1 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(192)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(192)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));//b

this->color2 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(170)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(170)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));//b

this->color3 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(210)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(210)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));//b

this->color4 = System::Drawing::Color::FromArgb(

static\_cast<System::Int32>(static\_cast<System::Byte>(200)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(200)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));//b

colorUpdate();

}

private: System::Void colorUpdate() {//функция обновления цветов приложения

this->BackColor = this->color1;

this->richTextBox1->BackColor = this->color2;

this->menuStrip1->BackColor = this->color3;

this->statusStrip1->BackColor = this->color4;

this->listBox1->BackColor = this->color2;

this->listBox2->BackColor = this->color2;

this->listBox3->BackColor = this->color2;

this->tabPage1->BackColor = this->color1;

this->tabPage2->BackColor = this->color1;

this->tabPage3->BackColor = this->color1;

}

private: System::Void debugSize(System::Object^ sender, System::EventArgs^ e) {//функция отладки (Не используется)

debug("off");

}

private: System::Void winResize(System::Object^ sender, System::EventArgs^ e) {//функция обработки изменения размера окна

//изменяем размеры элементов

this->richTextBox1->Height = this->size->TextH - (this->size->WinH - this->Height);

this->richTextBox1->Width = this->size->TextW - (this->size->WinW - this->Width);

this->tabControl1->Height = this->size->PanelH - (this->size->WinH - this->Height);

this->listBox1->Height = this->size->PanelListH - (this->size->WinH - this->Height);

this->listBox2->Height = this->listBox1->Height;

this->listBox3->Height = this->listBox1->Height;

this->tabControl1->Location = Point(this->size->PanelWPoint - (this->size->WinW - this->Width), this->tabControl1->Location.Y);

// debug(this->tabControl1->TabPages[0]->Height.ToString() + " " + this->tabControl1->TabPages[0]->Width.ToString());

}

private: System::Void debugLocation(System::Object^ sender, System::EventArgs^ e) {//функция отладки

String^ buf = "Win:\t" + this->Height + "\n\t" + this->Width + "\nText:\t" + this->richTextBox1->Height + "\n\t" + this->richTextBox1->Width + "\nPanel:\t" + this->tabControl1->Height + "\n\t" + this->tabControl1->Location.X + "p" + (this->size->PanelWPoint - (this->size->WinW - this->Width)).ToString();

MessageBox::Show(buf);

}

private: System::Void menuExit(System::Object^ sender, System::EventArgs^ e) {//функция выхода из приложения

this->Close();

}

private: System::Void menuFont(System::Object^ sender, System::EventArgs^ e) {//функция смены шрифта

this->textChangedEnable = false;//отключаем функцию изменения текста

this->fontDialog1->Font = this->richTextBox1->Font;//устанавливаем шрифт по умолчанию

this->fontDialog1->ShowDialog();//вызываем диалоговое окно

this->richTextBox1->Font = this->fontDialog1->Font;//применяем новый шрифт

this->textChangedEnable = true;//включаем функцию изменения текста

}

private: System::Void debugResize(System::Object^ sender, System::EventArgs^ e) {//функция отладки

this->Width = 360;

this->Height = 480;

}

private: bool colorOn = false;//параметр сброса цвета

private: System::Void menuAnalyzer(System::Object^ sender, System::EventArgs^ e) {//функция анализа программного кода

this->textChangedEnable = false;//отключаем функцию изменения текста

int bufIndex = this->richTextBox1->SelectionStart;//сохраняем положение курсора

this->funs = gcnew List<Ctring^>();//инициализируем список найденных функций

this->richTextBox1->SelectAll();//выделяем весь текст

this->richTextBox1->SelectionColor = System::Drawing::SystemColors::WindowText;//сбрасываем цвет

//очищаем текстбоксы

this->listBox1->Items->Clear();

this->listBox2->Items->Clear();

this->listBox3->Items->Clear();

Regex^ reg = gcnew Regex("[\\r\\n\\t :;]\*((~?[\_a-zA-Z][\_0-9a-zA-Z]+)[\\r\\n\\t ]\*\\([^)]\*\\))[\\r\\n\\t ]\*\\{[^}]\*\\}"); //задаем регекс строку для поиска функций

for each (Match ^ match in reg->Matches(this->richTextBox1->Text)) {//ищим функции

if (match->Groups[2]->Value != "each" &&

match->Groups[2]->Value != "while" &&

match->Groups[2]->Value != "catch" &&

match->Groups[2]->Value != "if" &&

match->Groups[2]->Value != "for") {//проверка на совпадения с ключевыми словами

Ctring^ buf = gcnew Ctring();//инициализируем новую функцию

buf->Text = match->Groups[1]->Value;//сохраняем название

buf->Index = match->Groups[2]->Index;//сохраняем указатель в тексте

this->funs->Add(buf);//добавляем в список

this->richTextBox1->Select(match->Groups[2]->Index, match->Groups[2]->Length);//выделяем название функции

this->richTextBox1->SelectionColor = System::Drawing::Color::FromArgb(//красим в красный цвет

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(0)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(0)));//b

}

}

this->funs->Sort();//сортируем в алфавитном порядке

for each (Ctring ^ a in this->funs) {//проходимся по списку функций и заносим их в листбокс

this->listBox2->Items->Add(a->Text);

}

reg = gcnew Regex("\\b((int|char|long|bool|wchar\_t|char16\_t|char32\_t|short|float|double|void)[\\r\\n\\t ]\*[\*|\\^]\*[\\r\\n\\t ]+[\_a-zA-Z][\_0-9a-zA-Z]\*)[\\r\\n\\t ]\*="); //устанавливаем регекс строку для поиска переменных

for each (Match ^ match in reg->Matches(this->richTextBox1->Text)) {//ищем переменные

this->listBox3->Items->Add(match->Groups[1]->Value);//добавляем переменную в листбокс

this->richTextBox1->Select(match->Groups[1]->Index, match->Groups[1]->Length);//выделяем переменную

this->richTextBox1->SelectionColor = System::Drawing::Color::FromArgb(//красим в фиолетовый

static\_cast<System::Int32>(static\_cast<System::Byte>(255)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(0)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));//b

}

for each (String ^ keyWord in initKeyword()) {//получаем ключевые слова из заранее заготовленного списка

reg = gcnew Regex("\\b" + keyWord + "\\b");//корректируем регекс строку для поиска ключевых слов

int buf = 0;//считаем их количество

for each (Match ^ match in reg->Matches(this->richTextBox1->Text)) {//ищем ключевые слова

this->richTextBox1->Select(match->Index, match->Length);//выделяем ключевые слова

this->richTextBox1->SelectionColor = System::Drawing::Color::FromArgb(//красим в синий цвет

static\_cast<System::Int32>(static\_cast<System::Byte>(0)), //r

static\_cast<System::Int32>(static\_cast<System::Byte>(0)), //g

static\_cast<System::Int32>(static\_cast<System::Byte>(255)));//b

buf++;//увеличиваем указатель на один

}

if (buf > 0) {//добавляем ключевое слово в список, если мы его нашли больше нуля раз.

this->listBox1->Items->Add(keyWord + " \t\t" + buf.ToString());

}

}

//красим листбоксы и включаем их

this->listBox1->ForeColor = System::Drawing::SystemColors::WindowText;

this->listBox2->ForeColor = System::Drawing::SystemColors::WindowText;

this->listBox3->ForeColor = System::Drawing::SystemColors::WindowText;

this->listBox1->Enabled = true;

this->listBox2->Enabled = true;

this->listBox3->Enabled = true;

this->richTextBox1->Select(bufIndex,0);//восстанавливаем положение курсора

this->colorOn = true;//устанавливаем разрешение на сброс цвета

this->textChangedEnable = true;//включаем функцию изменения текста

}

private: System::Void pickList2(System::Object^ sender, System::EventArgs^ e) {//функция выбора функции в списке

for each (Ctring ^ a in this->funs) {//проходимся по функциям в списке

if (this->listBox2->Items[this->listBox2->SelectedIndex]->ToString() == a->Text) {//ищем наш выбранную функцию

this->richTextBox1->Focus();//устанавливаем фокус на текстбокс

this->richTextBox1->SelectionStart = a->Index;//устанавливаем указатель на функцию

}

}

}

private: System::Void selectionChanged(System::Object^ sender, System::EventArgs^ e) {//функция обновления счетчика указателя в строке состояния

this->toolStripStatusLabel2->Text = "| " + this->richTextBox1->SelectionStart.ToString();

}

private: System::Void help(System::Object^ sender, System::EventArgs^ e) {//функция вызова справки

MessageBox::Show(this->TL3->Text, "", MessageBoxButtons::OK, MessageBoxIcon::Information, MessageBoxDefaultButton::Button1);

}

};

}

#pragma once

namespace CACPP {

using namespace System;

using namespace System::IO;

using namespace System::Collections::Generic;

ref class FileCPP

{

private:

String^ path;//путь файла

bool chanched = false;//статус файла

String^ text;//содержимое файла

public:

//Определение функций класса

FileCPP(String^ name);

void save();

property bool Status {//свойства статуса файла

bool get() { return chanched; }//вернуть состояние

void set(bool value) { chanched = value; }//установить состояния

}

property String^ Path {//свойства пути файла

String^ get() { return path; }//вернуть путь

void set(String^ value) { path = value; }//установить путь

}

property String^ Text {//свойства содержимого файла

String^ get() { return text; }//вернуть содержимое

void set(String^ value) { text = value; }//установить содержимое

}

};

}

#include "FileCPP.h"

namespace CACPP {

FileCPP::FileCPP(String^ name) {//инициализация класса

this->path = name;//сохраняем путь к файлу

this->text = gcnew String("");//инициализируем содержимое файла

if (path != "New") {//проверка, если файл новый

try {

StreamReader^ sr = gcnew StreamReader(this->path);//создаем поток чтения файла

String^ line = sr->ReadLine();//читаем первую строку

while (line != nullptr) {//продолжаем цикл, пока строка не пуста

this->text += (line + "\r\n");//сохраняем строку

line = sr->ReadLine();//читаем следующую линию

}

sr->Close();//закрываем поток

}

catch (FileNotFoundException^ ) {//исключение, если файл не обнаружен

//данное исключение не нарушает работу программы и должно игнорироваться

}

catch (UnauthorizedAccessException^ ) {//исключение на доступ к файлу

System::Windows::Forms::MessageBox::Show("Access denied, you must have administrator rights."); //в случаи исключения вызываем предупреждение

}

}

}

void FileCPP::save() {//сохранение файла

if (this->path == "New") {//Если файл новый, то его сохранять не нужно

return;

}

try {

StreamWriter^ sw = gcnew StreamWriter(this->path);//создаем поток записи файла

sw->Write(this->text);//записываем содержимое файла

sw->Close();//закрываем поток

}

catch (UnauthorizedAccessException^ ) {//исключение на доступ к файлу

System::Windows::Forms::MessageBox::Show("Access denied, you must have administrator rights."); //в случаи исключения вызываем предупреждение

}

}

}

#pragma once

namespace CACPP {

using namespace System;

using namespace System::Collections::Generic;

using namespace System::Text::RegularExpressions;

ref class Ctring : IComparable {//описываем класс и наследуем от класса IComparable для реализации сортировки

private:

String^ str;//название

int index;//индекс указателя

public:

//определяем функцию сравнения

int virtual CompareTo(Object^ obj) {

String^ str1 = ((Ctring^)obj)->Text;//Приводим тип к нашему классу для сравнения

return String::Compare(this->str, str1);//сравниваем строки по алфавитному порядку

}

property String^ Text {//свойства названия

String^ get() { return this->str; }//вернуть строку

void set(String^ value) { this->str = value; }//установить строку

}

property int Index {//свойство индекса указателя

int get() { return this->index; }//вернуть индекс

void set(int value) { this->index = value; }//установить индекс

}

};

List<String^>^ initKeyword() {//инициализируем список с ключевыми словами языка программирования С++

List<String^>^ buf = gcnew List<String^>();

buf->Add("alignas");

buf->Add("alignof");

buf->Add("and");

buf->Add("and\_eq");

buf->Add("asm");

buf->Add("auto");

buf->Add("bitand");

buf->Add("bitor");

buf->Add("bool");

buf->Add("break");

buf->Add("case");

buf->Add("catch");

buf->Add("char");

buf->Add("char8\_t");

buf->Add("char16\_t");

buf->Add("char32\_t");

buf->Add("class");

buf->Add("compl");

buf->Add("concept");

buf->Add("const");

buf->Add("const\_cast");

buf->Add("consteval");

buf->Add("constexpr");

buf->Add("constinit");

buf->Add("continue");

buf->Add("co\_await");

buf->Add("co\_return");

buf->Add("co\_yield");

buf->Add("decltype");

buf->Add("default");

buf->Add("delete");

buf->Add("do");

buf->Add("double");

buf->Add("dynamic\_cast");

buf->Add("else");

buf->Add("enum");

buf->Add("expicit");

buf->Add("export");

buf->Add("extern");

buf->Add("false");

buf->Add("float");

buf->Add("for");

buf->Add("friend");

buf->Add("goto");

buf->Add("if");

buf->Add("inline");

buf->Add("int");

buf->Add("long");

buf->Add("mutable");

buf->Add("namespace");

buf->Add("new");

buf->Add("noexcept");

buf->Add("not");

buf->Add("not\_eq");

buf->Add("nullptr");

buf->Add("operator");

buf->Add("or");

buf->Add("or\_eq");

buf->Add("private");

buf->Add("protected");

buf->Add("public");

buf->Add("register");

buf->Add("reinterpret\_cast");

buf->Add("requires");

buf->Add("return");

buf->Add("short");

buf->Add("signed");

buf->Add("sizeof");

buf->Add("static");

buf->Add("static\_assert");

buf->Add("static\_cast");

buf->Add("struct");

buf->Add("switch");

buf->Add("template");

buf->Add("this");

buf->Add("thread\_local");

buf->Add("throw");

buf->Add("true");

buf->Add("try");

buf->Add("typedef");

buf->Add("typeid");

buf->Add("typename");

buf->Add("union");

buf->Add("unsigned");

buf->Add("using");

buf->Add("virtual");

buf->Add("void");

buf->Add("volatile");

buf->Add("wchar\_t");

buf->Add("while");

buf->Add("xor");

buf->Add("xor\_eq");

return buf;

}

}

# Приложение Б (справочное) Результат работы программы

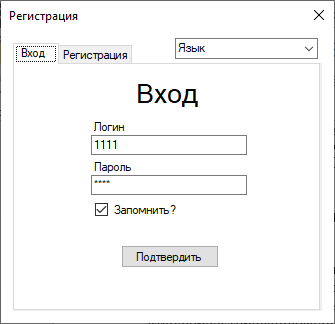


Рисунок Б.1 – Результат функции «Запустить приложение CACPP»



Рисунок Б.2 – Результат функции «Авторизоваться в приложении»

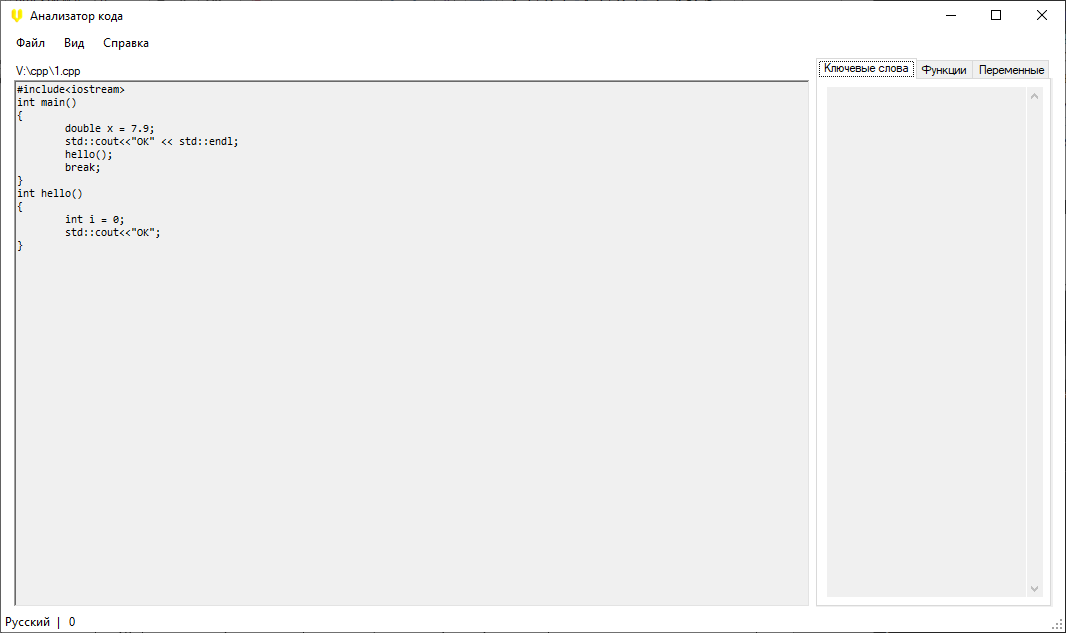


Рисунок Б.3 – Результат функции «Открыть файл»

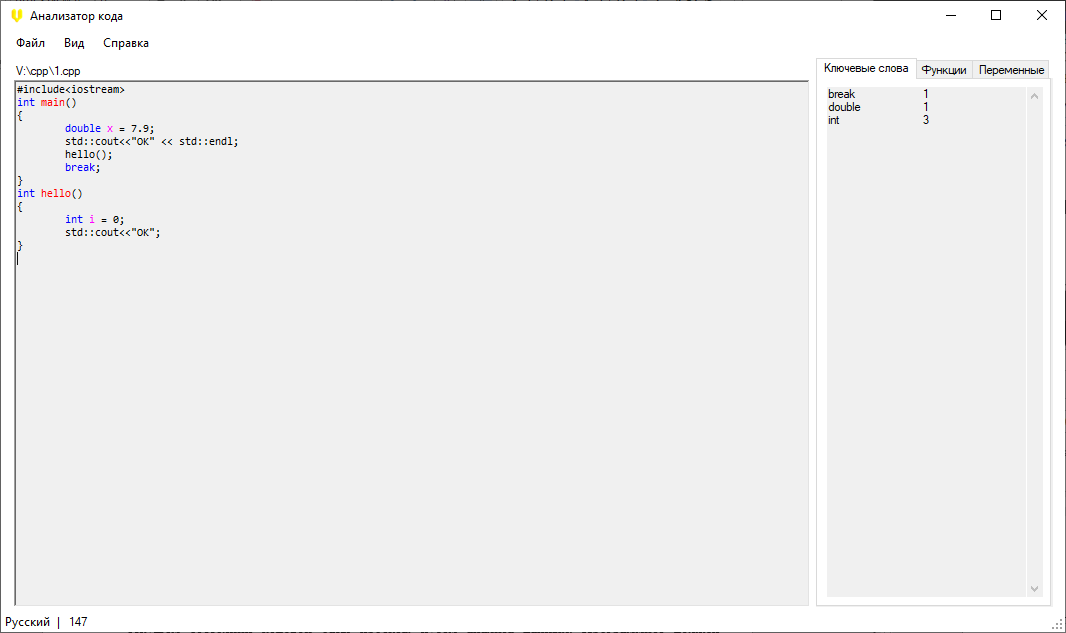


Рисунок Б.4 – Результат функции «Анализ программного кода»

# Приложение В (Обязательное) Блок-схемы работы алгоритмов

КБиП

У

Т. Контр.

Разраб.

Черняков В.А.

Провер.

Шаляпин Ю.В.

Реценз.

Утверд.

Н. Контр.

№ Докум.

Подпись

Дат

Изм.

Лист

Масса

Лит.

Масштаб

Лист 1

Листов 1

Инв.№подл.

Подп. и дата

Взам.инв.№в.

Инв.№ дубл.

Подп. и дата

КП Т.091020.401 ГЧ

КП Т.091020.401 ГЧ

Блок-схема входа в систему (общая)

*Программа для анализа программного кода на языке С++*

КБиП

У

Т. Контр.

Разраб.

Черняков В.А.

Провер.

Шаляпин Ю.В.

Реценз.

Утверд.

Н. Контр.

№ Докум.

Подпись

Дат

Изм.

Лист

Масса

Лит.

Масштаб

Лист 1

Листов 1

Инв.№подл.

Подп. и дата

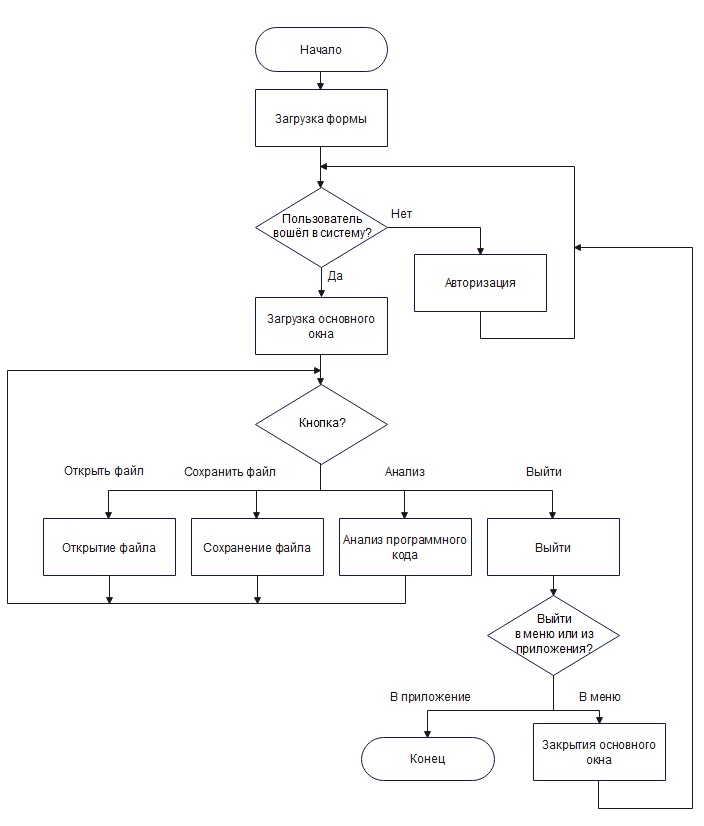
Взам.инв.№в.

Инв.№ дубл.

Подп. и дата

КП Т.091020.401 ГЧ

КП Т.091020.401 ГЧ



*Программа для анализа программного кода на языке С++*

Блок-схема работы программы (общая)

КБиП

У

Т. Контр.

Разраб.

Черняков В.А.

Провер.

Шаляпин Ю.В.

Реценз.

Утверд.

Н. Контр.

№ Докум.

Подпись

Дат

Изм.

Лист

Масса

Лит.

Масштаб

Лист 1

Листов 1

Инв.№подл.

Подп. и дата

Взам.инв.№в.

Инв.№ дубл.

Подп. и дата

КП Т.091020.401 ГЧ

КП Т.091020.401 ГЧ

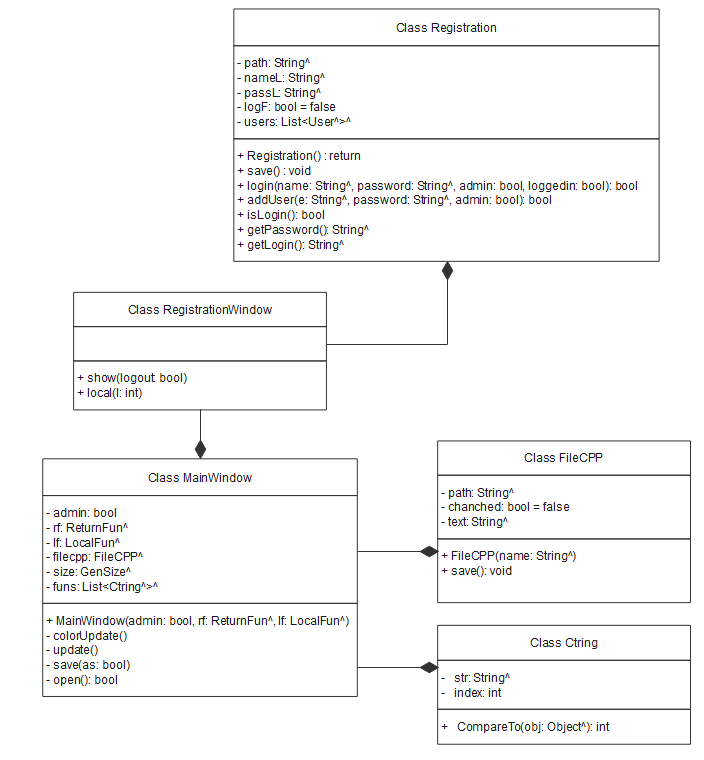


Диаграмма классов

(общая)

*Программа для анализа программного кода на языке С++*