Частное учреждение образование

«Колледж бизнеса и права»

ПРОГРАММА ДЛЯ НАХОЖДЕНИЯ НАИМЕНЬШЕГО ПУТИ МЕЖДУ ЗДАНИЯМИ В ГОРОДЕ МЕТОДОМ РАСЧЕТА МИНИМАЛЬНОГО ПУТИ МЕЖДУ ВЕРШИНАМИ ГРАФА ПО АЛГОРИТМУ ДЕЙКСТРЫ И

ВИЗУАЛИЗАЦИЯ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ И АЛГОРИТМА

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по учебному предмету

«Основы алгоритмизации и программирование»

КП Т.197018.404

Руководитель проекта (Ю.В. Шаляпин)

Обучающийся (И.А.Неворский)

2023

**СОДЕРЖАНИЕ**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

2

КП Т.197018.404 ПЗ

Разраб.

Неворский И.А.

Провер.

Шаляпин Ю.В.

Реценз.

Н. Контр.

Утверд.

*Программа для нахождения наименьшего пути между зданиями в городе методом расчета минимального пути между вершинами графа по алгоритму Дейкстры и визуализация полученных результатов и алгоритма*

Лит.

Листов

КБП

67

У

[Введение 3](#_Toc138801060)

[1.1 Сущность задачи 3](#_Toc138801061)

[1.2 Проектирование модели 5](#_Toc138801062)

[2 Вычислительная система 6](#_Toc138801063)

[2.1 Требование к аппаратным операционным ресурсам 6](#_Toc138801064)

[2.2 Инструменты разработки 6](#_Toc138801065)

[3 Проектирование задачи 8](#_Toc138801066)

[3.1 Требования к приложению 8](#_Toc138801067)

[3.2 Концептуальный прототип 8](#_Toc138801068)

[3.3 Организация данных 9](#_Toc138801069)

[3.4 Функции и элементы управления 9](#_Toc138801070)

[3.5 Проектирование справочной системы приложения 10](#_Toc138801071)

[4 Описание программного средства 11](#_Toc138801072)

[4.1 Общие сведения 11](#_Toc138801073)

[4.2 Входные и выходные данные 11](#_Toc138801074)

[5 Методика испытаний 12](#_Toc138801075)

[5.1 Технические требования 12](#_Toc138801076)

[5.2 Функциональное тестирование 12](#_Toc138801077)

[6 Применение 19](#_Toc138801078)

[6.1 Назначение программы 19](#_Toc138801079)

[6.2 Условия применения 19](#_Toc138801080)

[6.3 Справочная система 19](#_Toc138801081)

[Заключение 21](#_Toc138801082)

[Список использованных источников 22](#_Toc138801083)

[ПРИЛОЖЕНИЕ A 23](#_Toc138801084)

[ПРИЛОЖЕНИЕ Б 61](#_Toc138801085)

# Введение

Основной задачей данного проекта является разработка программного средства на языке программирования C++ с использованием среды разработки Microsoft Visual Studio для реализации функционала кратчайшего пути.

В первом разделе «Объектно– ориентированный анализ и проектирование системы» будет раскрыта организационная сущность задачи, предметная область и круг задач, подлежащих автоматизации. Также будут описаны основные функции программы и построена информационная модель, отражающая сущности задачи и их свойства.

Во втором разделе «Вычислительная система» будет представлено описание требований к аппаратным и операционным ресурсам, используемым инструментам разработки.

В третьем разделе «Проектирование задачи» будет содержаться описание приложения, включая требования к графическому дизайну, доступу к данным, ограничениям, возникающим при эксплуатации программы, а также текстовое и графическое описание пользовательского интерфейса, логическая и физическая структура данных, модули и функции, их взаимосвязь и разработка.

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

В пятом разделе «Методика испытаний» будут описаны требования к техническим средствам для проведения испытаний, функциональное тестирование, характеристики программы в условиях эксплуатации, а также требования к информационной и программной совместимости программного средства с областью его применения.

В шестом разделе «Применение» будет представлена структура справочной системы и методика ее использования.

В заключении будет проведен анализ созданного программного средства, оценена его соответствие поставленной задаче и выполненной работе.

## Сущность задачи

Целью является разработка на языке программирования С++ в среде разработки Microsoft Visual Studio программного средства для поиска кратчайшего пути. При поиске кратчайшего пути пользователи оперируют большим количеством информации, которая должна быть грамотно сгруппирована и понятно написана. Данная проблема на сегодняшний день очень обострена, так как система для занесения, хранения и представления информации должна быть единой для всех пользователей. Поэтому для достижения данной цели была поставлена и решена такая задача как: организованное занесение, хранение и отображение данных.

Поставленная задача может быть решена использованием компьютерных процессов обработки данных (информации). Программа способна быстро провести анализ занесенных данных и сообщить о кратчайшем пути между двумя точками, что значительно облегчает понимание информации пользователю.

Программы для поиска кратчайшего пути в основном применимы людьми, которые заняты в сферах деятельности, в которых используется информация. На данный момент реализовано большое количество программ аналогов таких как «Яндекс Навигатор», «Google Maps», «2GIS» и другие. Для сравнения мы рассмотрим программу Яндекс Навигатор. Данная программа реализует поиск кратчайшего пути между двумя точками. Функционал решения учитывает отраслевую специфику компаний, в сферу деятельности которых входит транспортировка грузов или людей.

Как пример похожих программ можно привести Яндекс Навигатор.

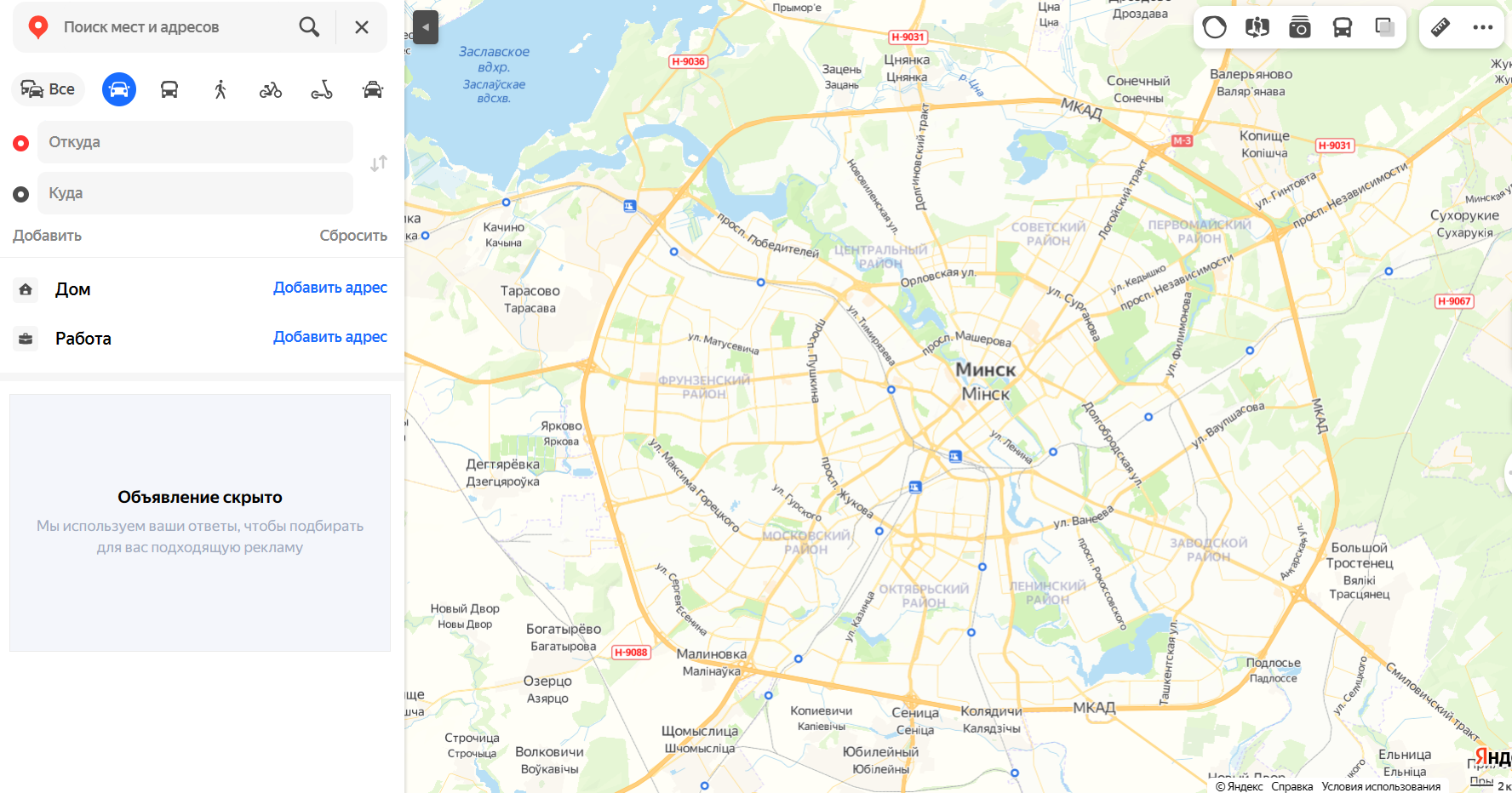


Рисунок .– Пример программы С++: Яндекс навигатор

Из достоинств программы «Яндекс Навигатор» можно выделить следующее: организованный и понятный интерфейс программы, наличие обширной функциональной базы.

Из недостатков программы «Яндекс Навигатор» можно выделить следующее: иногда маршруты могут быть неоптимальными, и могут быть проблемы с точностью определения местоположения.

Основным отличием разработанного ПС от аналог состоит в том, что разрабатываемое программное средство является более простым и менее требовательным в использовании. Разрабатываемая программа имеет удобный разработанный интерфейс, дружелюбную среду редактирования, ввода, хранения и вывода информации. Благодаря этому ПС не будет требовать огромных ресурсов операционный системы, но в тоже время реализовывать главный базовый функционал программного средства.

## 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 ли удачной регистрации, пользователь попадает в основное окно, где он может по своему усмотрению выполнить действия, а именно: сменить локализацию, цветовое оформление, получить справку, внести информация в базу данных, открыть основной файл, открыть справочный файл. Пользователь может комбинировать эти действия, получая функциональное приложение, способное удовлетворить все потребности пользователя.

Всё приложение основано на использовании файлов. Каждое действие имеет определенную логику, которая соответствует принципу работы с файлами. Все файлы, используемые в программном средстве, имеют определенные взаимосвязи.

# 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 ГБ;

– место на диске от 1 ГБ;

– клавиатура проводная;

– мышь проводная;

– видеопамять от 64 МБ.

## Инструменты разработки

Для разработки была задействована ОС Microsoft Windows 10 Pro. Языки программирования С++, С++/CLI. Инструменты и среды разработки Microsoft Visual Studio 2022, Windows Kits. 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 совместимости с Windows Form. Для инструментов разработки была использована среда MS Visual Studio 2022, имеющую поддержку разработки C++/CLI и Windows Form, также был выбран Windows Kits, позволяющий производить отладку приложения.

C++– компилируемый, статически типизированный язык программирования общего назначения. Поддерживает такие парадигмы программирования, как процедурное программирование, объектно– ориентированное программирование, обобщённое программирование.

Microsoft Visual Studio– линейка продуктов компании [Microsoft](https://ru.wikipedia.org/wiki/Microsoft), включающих [интегрированную среду разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) программного обеспечения и ряд других инструментальных средств.

MS Visual Studio включает в себя [редактор исходного кода](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BE%D0%B4%D0%B0) с поддержкой технологии [IntelliSense](https://ru.wikipedia.org/wiki/IntelliSense" \o "IntelliSense) и возможностью простейшего [рефакторинга кода](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3). Встроенный [отладчик](https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio_Debugger) может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб– редактор, дизайнер [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) и дизайнер [схемы базы данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%85%D0%B5%D0%BC%D0%B0_%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). MS Visual Studio позволяет создавать и подключать сторонние дополнения ([плагины](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%B3%D0%B8%D0%BD)) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем [контроля версий исходного кода](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D1%8F%D0%BC%D0%B8) (как, например, [Subversion](https://ru.wikipedia.org/wiki/Subversion" \o "Subversion) и [Visual SourceSafe](https://ru.wikipedia.org/wiki/Microsoft_Visual_SourceSafe)), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на [предметно– ориентированных языках программирования](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%B4%D0%BC%D0%B5%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA)) или инструментов для прочих аспектов [процесса разработки программного обеспечения](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F) (например, клиент Team Explorer для работы с [Team Foundation Server](https://ru.wikipedia.org/wiki/Team_Foundation_Server)).

MS Windows 10– проприетарная операционная система для персональных компьютеров, разработанная компанией Microsoft в рамках семейства Windows NT. Она является предшественником MS Windows 11. MS Windows 10 была представлена на мероприятии Microsoft 29 июля 2015 года и стала одной из самых популярных операционных систем в мире. Вместе с MS Windows 10 был выпущен Microsoft Office 2016, который предоставлял широкий спектр офисных приложений.

# 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.

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 3 до 15 символов.

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

## Концептуальный прототип

Интерфейс пользователя будет представлять из себя окно с левой частью, где расположены четыре текстовых поля для ввода информации о зданиях и расстояниях между ними. Первое текстовое поле предназначено для ввода номера первого здания, второе– для ввода номера второго здания, третье– для ввода расстояния между зданиями в метрах, а четвертое– для ввода необходимого количества зданий для работы программы. В правой части окна расположены два текстовых поля для ввода информации о начальном и конечном зданиях, а также три текстовых поля для вывода информации о кратчайшем пути. Первое текстовое поле будет содержать время для прохождения пути, второе– расстояние, а третье– порядок вершин, через которые необходимо идти для нахождения кратчайшего пути. Сверху должна располагаться строка меню, с выпадающими вкладками, позволяющее выбрать действие, такие как смена языка на русский, белорусский или английский, изменение цвета окна и справка. Снизу должен распалагаться рисунок графа.

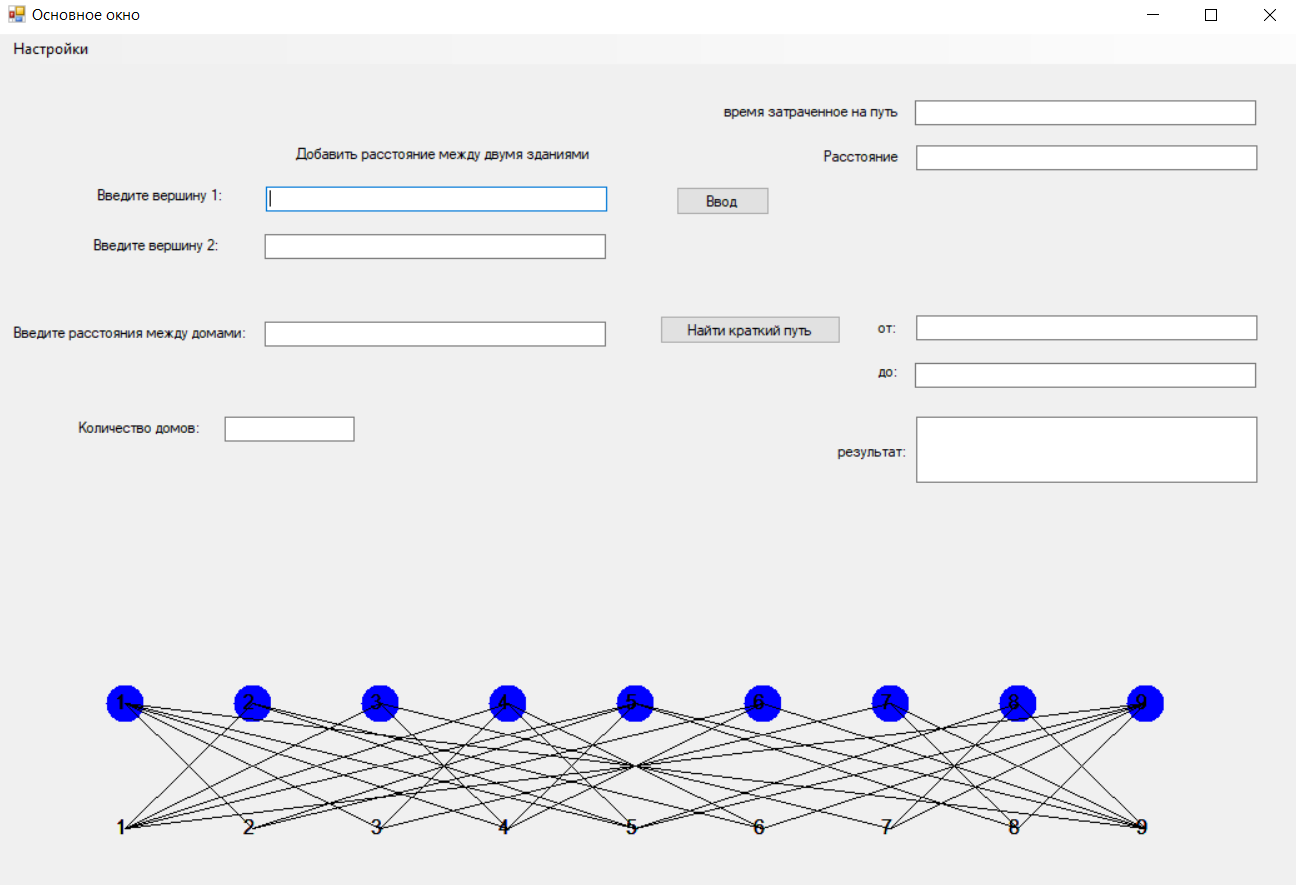


Рисунок .1– Концептуальный графический прототип интерфейса

## Организация данных

Логин и пароль пользователей, информация основного файла, информация справочного файла располагаются в файле по указанному пути «file\_with\_account.txt». 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 kursov, что позволит легко объединить функции в случае использования сторонних программных решений.

## 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 этого 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 функцию обработки определённого события.

Описание и текст всех функций приведён в приложении А.

## Проектирование справочной системы приложения

Справочная система реализована с помощью оконной формы Windows Form. Данный метод создания справочной системы позволяет легко и быстро выполнять функцию локализации (выбор языка зависит от пользователя) и перехода от одного окна ко второму. Всего в справочной системе программного средства реализовано 3 оконной формы Windows Form, каждая из которых предназначена для определеной оконной формы: первая оконная форма предназначена для авторизации, вторая– для регистрации, третья– для ввода информации о зданиях и их отображение в виде графа, затем можно использовать поиск пути. Каждая из 3 оконных форм имеет разъяснения по актуальным и более встречаемым вопросам.

Для вызова справочной системы используется элемент ToolStripMenuItem, который является встроенным элементом Windows Form.

# Описание программного средства

## Общие сведения

Приложение имеет название kursov. Данное название образованно от названия прокета в Microsofr Visual Studio 2022. Так же приложение имеет иконку в виде классисечкой иконки оконной формы.

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 4.7.2.

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.

## Входные и выходные данные

Разрабатываемая программа принимает входные данные в виде текста, который пользователь вводит в определённую оконную форму.

Для вывода данных в определённых оконных формах есть кнопки, которые отвечают за ввод данных на оконную форму.

Дополнительными данными является файл паролей и логинов пользователей, основной файл содержащий здания, файл– справочник. Пользователь не имеет прямого доступа к управлению этими файлами.

# 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 ГБ;

– место на диске от 1 ГБ;

– клавиатура проводная;

– мышь проводная;

– видеопамять от 64 МБ.

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.

Перед функциональным тестированием необходимо удостовериться в исправности работы базовой 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.

При запуске программного средства нам открывается приветственное окно, которое обладает двумя кнопками. Приветственное окно на рисунке 5.1.

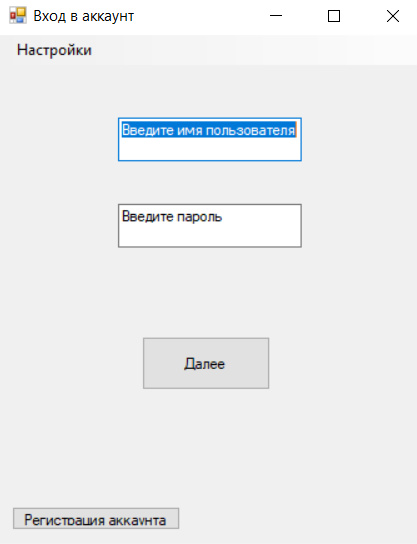


Рисунок 5.– Начальное окно

Для выполения программы нужно корректно ввести данные. Для входа в аккаунт нам следует нажать на кнопку «Далее». Если же у пользователя нету аккаунта, то следует нажать на элемент «Регистрация аккаунта», который показан на рисунке 5.2.



Рисунок 5.– Элемент дла перехода на окно «Регистрация аккаунта»

После нажатия на этот элемент будет позакано окно регистрация пользователя, которое показано на рисунке 5.3.

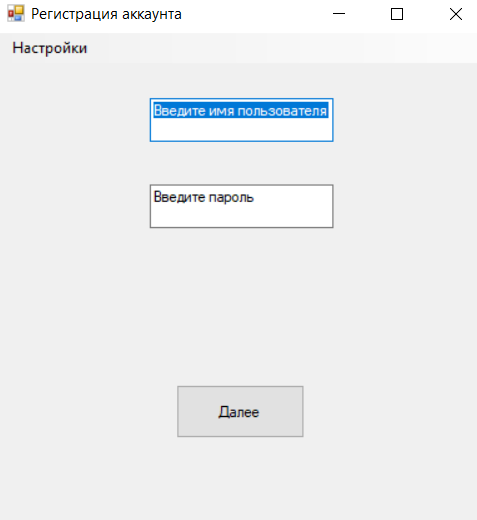


Рисунок 5.– Окно авторизации пользователя

Если вы ввели неверные данные в окне авторизации или в окне регистации, то будет выведено соответствующее сообщение об этом.

Если в окне авторизации Вы не ввели данные, то появится сообщение, которое показано на рисунке 5.4.



Рисунок 5.– Сообщения о неверно введённых данных в окне регистрации пользователя

Если в окне авторизации Вы ввели не правильный логин или пароль, которое показано на рисунке 5.5.



Рисунок 5.– Сообщение о неверно– введённых данных в окне авторизации пользователя

Если в окне регистрации Вы ввели данные, длина которых будет длиннее 15 символов, то появится сообщение, которое показано на рисунке 5.6.



Рисунок 5.– Сообщение о неверно– введённых данных в окне регистрации пользователя

Если в окне регистрации Вы ввели данные, длина которых будет менее 3 символов, то появится сообщение, которое показано на рисунке 5.7.



Рисунок 5.– Сообщение о неверно– введённых данных в окне регистрации пользователя

После введения данных (если данные введены верно), то Вы попадаете на основное окно приложения. Окно приложения показано на рисунке 5.8.

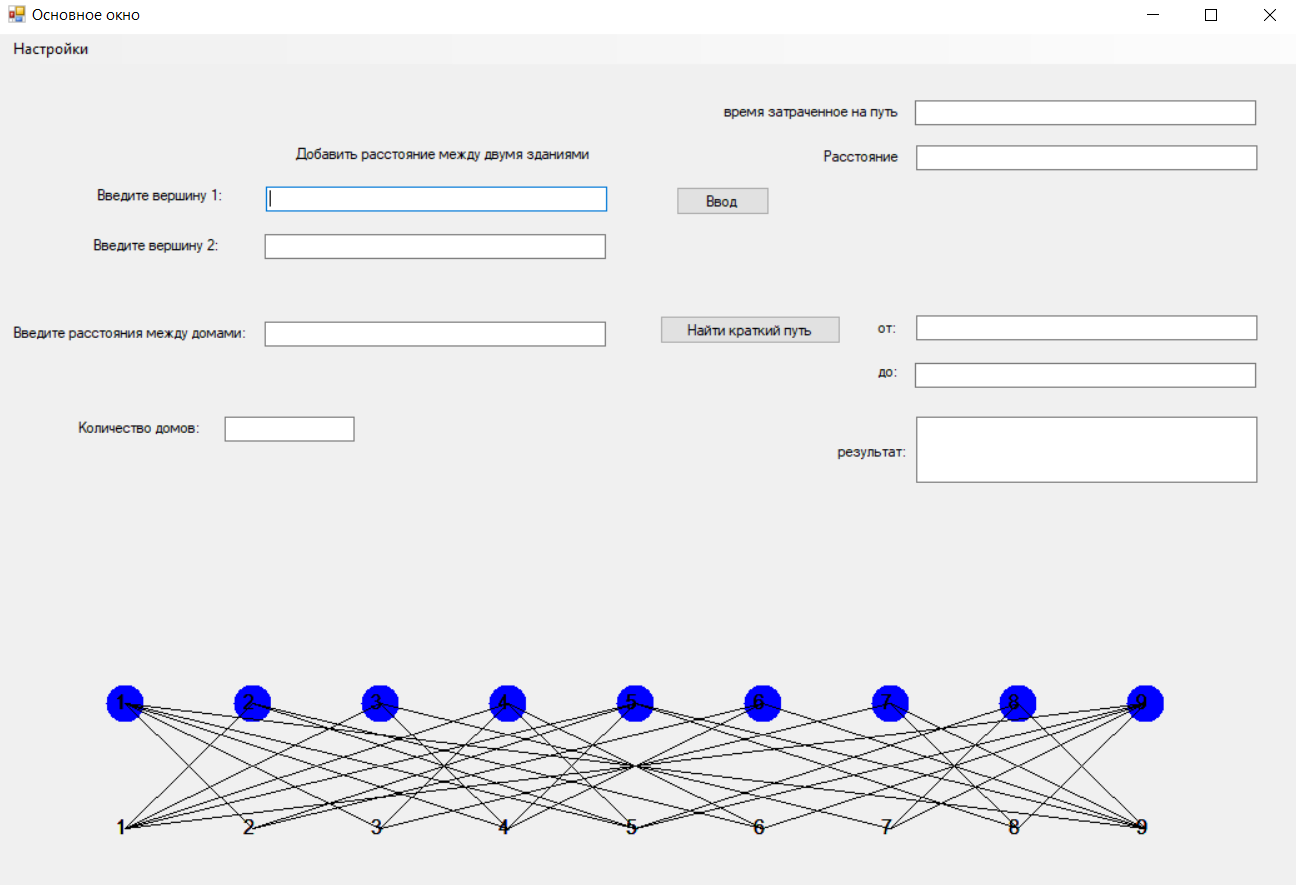


Рисунок 5.– Основное окно приложения

Вы должны ввести данные в необходимые поля. Если вы неверно ввели данные, то увидете соответствующее сообщение об этом.

Если Вы выберите одинаковые вершины, то Вам будет показано соответствующее сообщение. Сообщение показано на рисунке 5.9.



Рисунок 5.– Пример сообщения о неверном вводе данных

Если Вы введёте последнюю вершину больше чем существует, то вам будет выведено соответствующее сообщение. Сообщение показано на рисунке 5.10.



Рисунок 5.– Пример сообщения о неверном вводе данных

Если Вы верно введёте все данные, то они будут записаны в соответствующий файл. Запись в файл показана на рисунке 5.11.

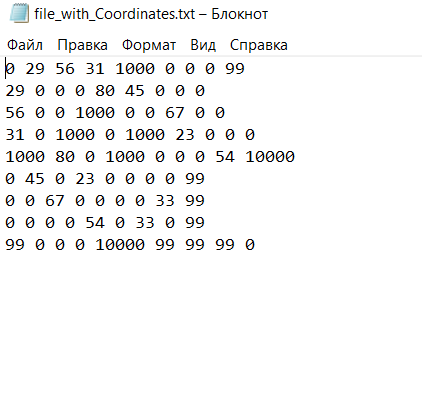


Рисунок 5.– Пример записанной информации в файл

Если Вы захотите добавить свзяь между домами или увеличить количество домов а затем это отрисовать, то Вы должны нажать на элемент «Ввод», который показан на рисунке 5.12.



Рисунок 5.– Кнопка для создании связи между домами и(или) добавлением дома а затем отрисовка графа

Если Вы захотите найти наикратчайший путь от одного дома до другого, то Вы должны нажать на элемент «Найти краткий путь», который показан на рисунке 5.13.



Рисунок 5.– Кнопка для нахождения кратчайшего пути

Вы должны ввести данные в необходимые поля. Если вы неверно ввели данные, то увидете соответствующее сообщение об этом.

Если Вы выберите одинаковые вершины, то Вам будет показано соответствующее сообщение. Сообщение показано на рисунке 5.14.



Рисунок 5.– Пример сообщения о неверном вводе данных

# Применение

## Назначение программы

Программа разработана с целью помочь программисту в решении задачи формирования представления о текущем состоянии кодовой базы проекта путем анализа данных. Она основана на принципах чтения и записи файлов. Основной задачей программы является разработка программного средства на языке программирования C++ в среде разработки Microsoft Visual Studio для реализации функциональности нахождения кратчайшего пути.

При работе с краткими путями программисты сталкиваются с обработкой большого объема информации, которую необходимо грамотно группировать и понятно представлять. Эта проблема актуальна, поскольку система для хранения, обработки и представления информации должна быть единообразной для всех пользователей программы, включая сотрудников различных организаций или проектов. Поэтому основной целью разработки данного программного средства является упрощение хранения и управления большим объемом информации о кратких путях.

Программа не предусматривает сетевую поддержку, так как она не требуется для нормального функционирования 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 ГБ;

– место на диске от 1 ГБ;

– клавиатура проводная;

– мышь проводная;

– видеопамять от 64 МБ;

– операционная система Microsoft Windows 10 pro.

Для применения разрабатываемого ПС необходимо установить Microsoft .Net Framework 4.7.2. Данный компонент будет предложено установить в ходе установки программы.

## Справочная система

Справочная система реализована с помощью оконной формы Windows Form. Данный метод создания справочной системы позволяет легко и быстро выполнять функцию локализации (выбор языка зависит от пользователя) и перехода от одного окна ко второму. Всего в справочной системе программного средства реализовано 3 оконной формы Windows Form, каждая из которых предназначена для определеной оконной формы: первая оконная форма предназначена для авторизации, вторая– для регистрации, третья– для ввода информации о зданиях и их отображение в виде графа, затем можно использовать поиск пути. Каждая из 3 оконных форм имеет разъяснения по актуальным и более встречаемым вопросам.

Для вызова справочной системы используется элемент ToolStripMenuItem, который является встроенным элементом Windows Form.

# Заключение

Были разработаны и созданы базовые компоненты программного обес– печения, реализующего программу для нахождения краткого пути.

Для разработки ПС и достижения поставленной задачи были освоены знания по предметной области «Основы алгоритмизации и программирова– ния».

В ходе разработки ПС были применены различные программные и ап– паратные решения. Были изучены основы Microsoft .Net Framework 4.7.2 в связке с Windows Form и Windows API. Были изучены основы синтаксиса С++ и все исключающие возможности языка для корректного расчета кратких путей. Также был применен весь ранее изученный материал данной дисциплины. Освоена технология создания справки средствами Windows Form и CHM– файлов.

За все время разработки было использовано большое количество фай– лов, которые помогли вспомнить принципы работы с файлами. Также были закреплены основы работы в среде разработки Microsoft Visual Studio 2022.

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

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

В программе реализованы такие задачи как:

– авторизация в системе пользователя;

– изменение, создание, чтение и запись файлов;

– вывод информации.

Были реализованы такие требования к приложению как локализация программы на трех языках, изменение цветового оформления заднего фона, удобная справочная система.

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

Преимуществами данного ПС являются: относительная легкость установки программы, маленький вес программы, доступность локализации, простота функционала, устройствонезависимость.

Недостатками данного ПС являются ограниченный функционал и отсут– ствие кроссплатформенности.

Разработка и сопровождение данного ПС реализованы должным обра– зом. Разработанное ПС полностью отлажено. Было проведено полное мануальное тестирование.

Поставленные задачи выполнены. Разработанное ПС является готовым продуктом, годным к использованию.

# Список использованных источников

1. Багласова, Т.Г. Методические указания по оформлению курсовых проектов, дипломных проектов и отчетов для учащихся специальности 2– 40 01 01 «Программное обеспечение информационных технологий» // Т.Г. Багласова, К.О. Якимович и др.– Минск : ЧУО «КБП» , 2022 .– 41 с.
2. Дейтел, Х. Как программировать на C++ / Х. Дейтел, П. Дейтел.– М. : ДМК Пресс , 2018 .– 1021 с.
3. Каталог API (Microsoft) материалов [Электронный ресурс].– Microsoft , 2023– Режим доступа : https://docs.microsoft.com/en– us/windows/win32/appindex/windows– api– list.– Дата доступа : 24.05.2023.
4. Мейерс, С.Эффективное использование С++. 55 верных способов улучшить структуру и код ваших программ / С. Мейерс. – М.: ДМК Прессь,ь2006 .– 300 с.
5. Общие требования к тестовым документам : ГОСТ 2.105– 95.– Введ. 01.01.1996 .–  Минск : Межгос. совет по стандартизации, метрологии и сертификации , 1995 .– 84 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. 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.
8. Программа и методика испытаний. Требования к содержанию, оформлению и контролю качества : ГОСТ 19.301– 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 19.404– 2000 .– Введ. 01.09.2001 .– Минск : Межгос. совет по стандартизации, метрологии и сертификации , 2000 .– 16 с.
10. Шаляпин, Ю.В. Основы алгоритмизации и программирование : методические указания к курсовому проектированию для учащихся специальности 2– 40 01 01 «Программное обеспечение информационных технологий» / Ю.В. Шаляпин. Минск : ЧУО «КБП» , 2022 .– 19 с.

# ПРИЛОЖЕНИЕ А (обязательное) Текст программы

//Файл RegistrationWindow.cpp

#include “RegistrationWindow.h”

#include <Windows.h>

using namespace kursov;

[STAThreadAttribute]

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)

{

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Application::Run(gcnew RegistrationWindow());

return 0;

}

//Файл CreateAccountWindow.cpp

#include “CreateAccountWindow.h”

//Файл MainWindow.cpp

#include “MainWindow.h”

//Файл RegistrationWindow.h

#pragma endregion

String^ filename = “file\_with\_account.txt”;

bool flag\_for\_textbox = false;

int language\_now = 0;//0– рус, 1– англ, 2– бел

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e)

{

//регистрация нового аккаунта(открывает новое окно)

CreateAccountWindow^ createwindow = gcnew CreateAccountWindow(language\_now);

createwindow– >Show();

this– >Hide();

}

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

{

String^ line;

String^ login;

String^ password;

String^ login\_input = this– >textBox1– >Text;

String^ password\_input = this– >textBox2– >Text;

bool flag = false;

try

{

// Проверяем, существует ли файл

if (!File::Exists(filename))

{

// Если файл не существует, создаем его

File::Create(filename);

}

if (String::IsNullOrEmpty(login\_input) || String::IsNullOrEmpty(password\_input))

{

// Выводим сообщение об ошибке в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Логин и пароль не должны быть пустыми”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “Login and password should not be empty”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Лагін і пароль не павінны быць пустымі”;

}

return; // Выходим из метода

}

// Проверяем длину вводимого логина и пароля

if (login\_input– >Length < 3 || login\_input– >Length > 15)

{

// Выводим сообщение об ошибке в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Длина логина должна быть от 3 до 15 символов”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “The length of the login should be between 3 and 15 characters”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Даўжыня лагіна павінна быць ад 3 да 15 сімвалаў”;

}

return; // Выходим из метода

}

if (password\_input– >Length < 3 || password\_input– >Length > 15)

{

// Выводим сообщение об ошибке в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Длина пароля должна быть от 3 до 15 символов”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “The length of the password should be between 3 and 15 characters”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Даўжыня пароля павінна быць ад 3 да 15 сімвалаў”;

}

return; // Выходим из метода

}

// Проверяем наличие знака “:” в логине или пароле

if (login\_input– >Contains(“:”) || password\_input– >Contains(“:”))

{

// Выводим сообщение об ошибке в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Логин и пароль не должны содержать символ ':'“;

}

else if (language\_now == 1)

{

this– >label1– >Text = “Login and password should not contain the ':' character”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Лагін і пароль не павінны змяшчаць сімвал ':'“;

}

return; // Выходим из метода

}

// Создаем объект для чтения файла

StreamReader^ reader = gcnew StreamReader(filename);

// Читаем файл построчно

while ((line = reader– >ReadLine()) != nullptr)

{

// Разделяем строку на логин и пароль по символу “:”

array<String^>^ tokens = line– >Split(':');

// Проверяем, что строка содержит логин и пароль

if (tokens– >Length != 2)

{

// Пропускаем некорректные строки

continue;

}

// Извлекаем логин и пароль из массива

login = tokens[0];

password = tokens[1];

// Проверяем, совпадают ли введенные логин и пароль с данными из файла

if (login\_input == login && password\_input == password)

{

// Если логин и пароль совпадают, устанавливаем флаг в true и выходим из цикла

flag = true;

break;

}

else

{

// Если логин и пароль не совпадают, продолжаем поиск

continue;

}

}

// Закрываем объект для чтения файла

reader– >Close();

// Проверяем значение флага

if (flag == false)

{

// Если флаг равен false, выводим сообщение об ошибке в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Проверьте логин или пароль, они введены некорректно”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “Check your login or password, they are entered incorrectly”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Праверце лагін ці пароль, яны уведзены няправільна”;

}

}

else

{

// Если флаг равен true, выводим сообщение об успешном входе в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Вход выполнен”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “Login successful”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Уваход выкананы”;

}

// Создаем объект главного окна

MainWindow^ mainwindow = gcnew MainWindow(language\_now);

// Показываем главное окно

mainwindow– >Show();

// Скрываем текущее окно

this– >Hide();

}

}

catch (...)

{

// Обработка ошибок при записи данных или их проверке

if (language\_now == 0)

{

Console::WriteLine(“Ошибка при записи данных или при их проверке”);

//вывод ошибки

}

else if (language\_now == 1)

{

Console::WriteLine(“Error writing or validating data”);

//вывод ошибки

}

else if (language\_now == 2)

{

Console::WriteLine(“Памылка пры запісе або праверцы дадзеных”);

//вывод ошибки

}

}

}

private: System::Void Textbox1\_clear(System::Object^ sender, System::EventArgs^ e)

{

//очистка текстбоксов при их нажатии

try

{

if (flag\_for\_textbox == false)

{

textBox1– >Clear();

textBox2– >Clear();

flag\_for\_textbox = true;

}

}

catch (...)

{

if (language\_now == 0)

{

MessageBox::Show(“Ошибка при очистке textbox”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Error clearing textbox”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Ошибка при очистке textbox”);

}

}

}

private: System::Void Textbox2\_clear(System::Object^ sender, System::EventArgs^ e)

{

//очистка текстбоксов при их нажатии

try

{

if (flag\_for\_textbox == false)

{

textBox1– >Clear();

textBox2– >Clear();

flag\_for\_textbox = true;

}

}

catch (...)

{

if (language\_now == 0)

{

MessageBox::Show(“Ошибка при очистке textbox”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Error clearing textbox”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Ошибка при очистке textbox”);

}

}

}

private: System::Void бирюзовыйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::Turquoise; //для берюзового цвета

}

private: System::Void красныйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::Red; //для красного цвета

}

private: System::Void жёлтыйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::Yellow; //для желтого цвета

}

private: System::Void белыйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::White; //для белого цвета

}

private: System::Void англToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

language\_now = 1;

this– >Text = “Log in”;

this– >button1– >Text = “Next”;

this– >button2– >Text = “Register account”;

this– >textBox1– >Text = “Enter username”;

this– >textBox2– >Text = “Enter password”;

flag\_for\_textbox = false;

this– >бирюзовыйToolStripMenuItem– >Text = “Turquoise”;

this– >красныйToolStripMenuItem– >Text = “Red”;

this– >жёлтыйToolStripMenuItem– >Text = “Yellow”;

this– >белыйToolStripMenuItem– >Text = “White”;

this– >цветОкнаToolStripMenuItem– >Text = “Window color”;

this– >языкToolStripMenuItem– >Text = “Language”;

this– >настройкиToolStripMenuItem– >Text = “Settings”;

this– >англToolStripMenuItem– >Text = “English”;

this– >белорсукийToolStripMenuItem– >Text = “Belarusian”;

this– >русскийToolStripMenuItem– >Text = “Russian”;

}

private: System::Void белоруccкийToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

language\_now = 2;

this– >Text = “Уваход у акаўнт”;

this– >button1– >Text = “Далей”;

this– >button2– >Text = “Рэгістрацыя акаўнта”;

this– >textBox1– >Text = “Увядзіце імя карыстальніка”;

this– >textBox2– >Text = “Увядзіце пароль”;

flag\_for\_textbox = false;

this– >бирюзовыйToolStripMenuItem– >Text = “Бірюзавы”;

this– >красныйToolStripMenuItem– >Text = “Чырвоны”;

this– >жёлтыйToolStripMenuItem– >Text = “Жоўты”;

this– >белыйToolStripMenuItem– >Text = “Белы”;

this– >цветОкнаToolStripMenuItem– >Text = “Колір акна”;

this– >языкToolStripMenuItem– >Text = “Мова”;

this– >настройкиToolStripMenuItem– >Text = “Настройкі”;

this– >англToolStripMenuItem– >Text = “Англійская”;

this– >белорсукийToolStripMenuItem– >Text = “Беларуская”;

this– >русскийToolStripMenuItem– >Text = “Руская”;

}

private: System::Void русскийToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

language\_now = 0;

this– >Text = “Вход в аккаунт”;

this– >button1– >Text = “Далее”;

this– >button2– >Text = “Регистрация аккаунта”;

this– >textBox1– >Text = “Введите имя пользователя”;

this– >textBox2– >Text = “Введите пароль”;

flag\_for\_textbox = false;

this– >бирюзовыйToolStripMenuItem– >Text = “Бирюзовый”;

this– >красныйToolStripMenuItem– >Text = “Красный”;

this– >жёлтыйToolStripMenuItem– >Text = “Жёлтый”;

this– >белыйToolStripMenuItem– >Text = “Белый”;

this– >цветОкнаToolStripMenuItem– >Text = “Цвет окна”;

this– >языкToolStripMenuItem– >Text = “Язык”;

this– >настройкиToolStripMenuItem– >Text = “Настройки”;

this– >англToolStripMenuItem– >Text = “Английский”;

this– >белорсукийToolStripMenuItem– >Text = “Белорсукий”;

this– >русскийToolStripMenuItem– >Text = “Русский”;

}

};

}

//Файл CreateAccountWindow.h

#pragma endregion

String^ filename = “file\_with\_account.txt”;

bool flag\_for\_textbox = false;

private: System::Void Textbox1\_clear(System::Object^ sender, System::EventArgs^ e)

{

//очистка текстбоксов при их нажатии

try

{

if (flag\_for\_textbox == false)

{

textBox1– >Clear();

textBox2– >Clear();

flag\_for\_textbox = true;

}

}

catch (...)

{

if (language\_now == 0)

{

MessageBox::Show(“Ошибка при очистке textbox”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Error clearing textbox”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Ошибка при очистке textbox”);

}

}

}

private: System::Void Textbox2\_clear(System::Object^ sender, System::EventArgs^ e)

{

//очистка текстбоксов при их нажатии

try

{

if (flag\_for\_textbox == false)

{

textBox1– >Clear();

textBox2– >Clear();

flag\_for\_textbox = true;

}

}

catch (...)

{

if (language\_now == 0)

{

MessageBox::Show(“Ошибка при очистке textbox”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Error clearing textbox”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Ошибка при очистке textbox”);

}

}

}

private: System::Void check\_and\_creat(System::Object^ sender, System::EventArgs^ e)

{

String^ line;

String^ login;

String^ password;

String^ login\_input = this– >textBox1– >Text;

String^ password\_input = this– >textBox2– >Text;

bool flag = false;

try

{

// Проверяем, существует ли файл

if (!File::Exists(filename))

{

// Если файл не существует, создаем его

File::Create(filename);

}

if (String::IsNullOrEmpty(login\_input) || String::IsNullOrEmpty(password\_input))

{

// Выводим сообщение об ошибке в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Логин и пароль не должны быть пустыми”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “Login and password should not be empty”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Лагін і пароль не павінны быць пустымі”;

}

return; // Выходим из метода

}

// Проверяем длину вводимого логина и пароля

if (login\_input– >Length < 3 || login\_input– >Length > 15)

{

// Выводим сообщение об ошибке в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Длина логина должна быть от 3 до 15 символов”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “The length of the login should be between 3 and 15 characters”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Даўжыня лагіна павінна быць ад 3 да 15 сімвалаў”;

}

return; // Выходим из метода

}

if (password\_input– >Length < 3 || password\_input– >Length > 15)

{

// Выводим сообщение об ошибке в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Длина пароля должна быть от 3 до 15 символов”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “The length of the password should be between 3 and 15 characters”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Даўжыня пароля павінна быць ад 3 да 15 сімвалаў”;

}

return; // Выходим из метода

}

// Проверяем наличие знака “:” в логине или пароле

if (login\_input– >Contains(“:”) || password\_input– >Contains(“:”))

{

// Выводим сообщение об ошибке в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Логин и пароль не должны содержать символ ':'“;

}

else if (language\_now == 1)

{

this– >label1– >Text = “Login and password should not contain the ':' character”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Лагін і пароль не павінны змяшчаць сімвал ':'“;

}

return; // Выходим из метода

}

// Бесконечный цикл

while (true)

{

// Создаем объект для чтения файла

StreamReader^ reader = gcnew StreamReader(filename);

// Читаем файл построчно

while ((line = reader– >ReadLine()) != nullptr)

{

// Разделяем строку на логин и пароль по символу “:”

array<String^>^ tokens = line– >Split(':');

// Проверяем, что строка содержит логин и пароль

if (tokens– >Length != 2)

{

// Пропускаем некорректные строки

continue;

}

// Извлекаем логин и пароль из массива

login = tokens[0];

password = tokens[1];

// Проверяем, совпадают ли введенные логин и пароль с данными из файла

if (login\_input == login)

{

// Если логин и пароль совпадают, устанавливаем флаг в true и выходим из цикла

flag = true;

break;

}

else

{

flag = false;

continue;

}

}

// Закрываем объект для чтения файла

reader– >Close();

// Проверяем значение флага

if (flag == false)

{

// Если флаг равен false, создаем объект для записи в файл

StreamWriter^ sw = gcnew StreamWriter(filename, true); // дозапись

// Записываем логин и пароль в файл

sw– >Write(login\_input + “:” + password\_input + “\n”);

// Закрываем объект для записи

sw– >Close();

// Выводим сообщение об успешном создании аккаунта в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Аккаунт создан”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “Account created”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Рахунак створаны”;

}

MainWindow^ f = gcnew MainWindow(language\_now);

f– >Show();

this– >Hide();

break;

}

else

{

// Если флаг равен true, выводим сообщение о существующем аккаунте в зависимости от текущего языка

if (language\_now == 0)

{

this– >label1– >Text = “Такой аккаунт уже существует, придумайте другой”;

}

else if (language\_now == 1)

{

this– >label1– >Text = “Such an account already exists, come up with another one”;

}

else if (language\_now == 2)

{

this– >label1– >Text = “Такі акаўнт ужо існуе, выдумайце іншы”;

}

break;

}

}

}

catch (...)

{

// Обработка ошибок при записи данных или их проверке

if (language\_now == 0)

{

Console::WriteLine(“Ошибка при записи данных или при их проверке”);

}

else if (language\_now == 1)

{

Console::WriteLine(“Error writing or validating data”);

}

else if (language\_now == 2)

{

Console::WriteLine(“Памылка пры запісе або праверцы дадзеных”);

}

}

}

private: System::Void бирюзовыйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::Turquoise; //для берюзового цвета

}

private: System::Void красныйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::Red; //для красного цвета

}

private: System::Void жёлтыйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::Yellow; //для желтого цвета

}

private: System::Void белыйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::White; //для белого цвета

}

private: System::Void англToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >Text = “Registration of an account”;

language\_now = 1;

this– >button1– >Text = “Next”;

this– >textBox1– >Text = “Enter username”;

this– >textBox2– >Text = “Enter password”;

flag\_for\_textbox = false;

this– >бирюзовыйToolStripMenuItem– >Text = “Turquoise”;

this– >красныйToolStripMenuItem– >Text = “Red”;

this– >жёлтыйToolStripMenuItem– >Text = “Yellow”;

this– >белыйToolStripMenuItem– >Text = “White”;

this– >цветОкнаToolStripMenuItem– >Text = “Window color”;

this– >языкToolStripMenuItem– >Text = “Language”;

this– >настройкиToolStripMenuItem– >Text = “Settings”;

this– >англToolStripMenuItem– >Text = “English”;

this– >белорсукийToolStripMenuItem– >Text = “Belarusian”;

this– >русскийToolStripMenuItem– >Text = “Russian”;

}

private: System::Void белоруccкийToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >Text = “Рэгістрацыя ўліковага запісу”;

language\_now = 2;

this– >button1– >Text = “Далей”;

this– >textBox1– >Text = “Увядзіце імя карыстальніка”;

this– >textBox2– >Text = “Увядзіце пароль”;

flag\_for\_textbox = false;

this– >бирюзовыйToolStripMenuItem– >Text = “Бірюзавы”;

this– >красныйToolStripMenuItem– >Text = “Чырвоны”;

this– >жёлтыйToolStripMenuItem– >Text = “Жоўты”;

this– >белыйToolStripMenuItem– >Text = “Белы”;

this– >цветОкнаToolStripMenuItem– >Text = “Колір акна”;

this– >языкToolStripMenuItem– >Text = “Мова”;

this– >настройкиToolStripMenuItem– >Text = “Настройкі”;

this– >англToolStripMenuItem– >Text = “Англійская”;

this– >белорсукийToolStripMenuItem– >Text = “Беларуская”;

this– >русскийToolStripMenuItem– >Text = “Руская”;

}

private: System::Void русскийToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >Text = “Регистрация аккаунта”;

language\_now = 0;

this– >button1– >Text = “Далее”;

this– >textBox1– >Text = “Введите имя пользователя”;

this– >textBox2– >Text = “Введите пароль”;

flag\_for\_textbox = false;

this– >бирюзовыйToolStripMenuItem– >Text = “Бирюзовый”;

this– >красныйToolStripMenuItem– >Text = “Красный”;

this– >жёлтыйToolStripMenuItem– >Text = “Жёлтый”;

this– >белыйToolStripMenuItem– >Text = “Белый”;

this– >цветОкнаToolStripMenuItem– >Text = “Цвет окна”;

this– >языкToolStripMenuItem– >Text = “Язык”;

this– >настройкиToolStripMenuItem– >Text = “Настройки”;

this– >англToolStripMenuItem– >Text = “Английский”;

this– >белорсукийToolStripMenuItem– >Text = “Белорсукий”;

this– >русскийToolStripMenuItem– >Text = “Русский”;

}

};

}

//Файл MainWindow.h

void DrawGraph(Graphics^ g, int\*\* graph, int size)

{

// Определите параметры отображения графа, такие как размер узлов и цвета

int nodeSize = 30;

Color nodeColor = Color::Blue;

Color edgeColor = Color::Black;

Pen^ pen = gcnew Pen(edgeColor);

SolidBrush^ brush = gcnew SolidBrush(nodeColor);

// Вычислите расстояние между узлами, исходя из размера панели рисования

int distance = g– >VisibleClipBounds.Width / (size + 1);

// Отобразите узлы графа

for (int i = 0; i < size; i++)

{

int x = (i + 1) \* distance; // Вычислите координаты узла

int y = 150; // Фиксированная координата узла по оси Y

g– >FillEllipse(brush, x– nodeSize / 2, y– nodeSize / 2, nodeSize, nodeSize); // Нарисуйте узел

g– >DrawString((i + 1).ToString(), gcnew System::Drawing::Font(“Arial”, 12), Brushes::Black, x– 10, y– 10);

}

// Отобразите связи между узлами

for (int i = 0; i < size; i++)

{

for (int j = 0; j < size; j++)

{

if (graph[i][j] > 0)

{

int x1 = (i + 1) \* distance; // Вычислите координаты начального узла связи

int y1 = 150; // Фиксированная координата начального узла по оси Y

int x2 = (j + 1) \* distance; // Вычислите координаты конечного узла связи

int y2 = 250; // Фиксированная координата конечного узла по оси Y

g– >DrawLine(pen, x1, y1, x2, y2); // Нарисуйте связь

g– >DrawString((i + 1).ToString(), gcnew System::Drawing::Font(“Arial”, 12), Brushes::Black, x1– 10, y1– 10);

g– >DrawString((j + 1).ToString(), gcnew System::Drawing::Font(“Arial”, 12), Brushes::Black, x2– 10, y2– 10);

}

}

}

}

void dijkstra(int\*\* a, int SIZE, int begin\_index, int end\_index)

{

// Очистить текстовые поля перед запуском алгоритма

this– >textBox4– >Clear();

this– >textBox5– >Clear();

this– >textBox6– >Clear();

const float speed = 1.5; // Скорость движения (км/ч)

long long\* d = new long long[SIZE]; // Массив для хранения минимальных расстояний

long long\* v = new long long[SIZE]; // Массив для отметки посещенных узлов

long long temp, minindex, min;

// Инициализация массивов

for (int i = 0; i < SIZE; i++) {

d[i] = 9223372036854775807; // Задать начальное расстояние как максимальное

v[i] = 1; // Отметить все узлы как непосещенные

}

d[begin\_index] = 0; // Начальное расстояние до начального узла равно 0

do {

minindex = 9223372036854775807;

min = 9223372036854775807;

// Найти узел с минимальным расстоянием из непосещенных узлов

for (int i = 0; i < SIZE; i++) {

if ((v[i] == 1) && (d[i] < min)) {

min = d[i];

minindex = i;

}

}

// Если найденный узел не равен максимальному значению

if (minindex != 9223372036854775807) {

// Обновить расстояния до соседних узлов через текущий узел

for (int i = 0; i < SIZE; i++) {

if (a[minindex][i] > 0) {

temp = min + a[minindex][i];

if (temp < d[i]) {

d[i] = temp;

}

}

}

v[minindex] = 0; // Пометить узел как посещенный

}

} while (minindex < 9223372036854775807);

long long\* ver = new long long[SIZE]; // Массив для хранения пути

ver[0] = end\_index; // Начать с конечного узла

int k = 1;

int weight = d[end\_index]; // Получить общее расстояние пути

this– >textBox6– >Text = Convert::ToString(weight) + “ m. ( “ + Convert::ToString(weight / 1000.0) + “ км. )”;

// Рассчитать время в секундах и преобразовать в формат часы:минуты:секунды

int timeInSeconds = static\_cast<int>(weight / speed);

int hours = timeInSeconds / 3600;

int minutes = (timeInSeconds % 3600) / 60;

int seconds = timeInSeconds % 60;

this– >textBox5– >Text = hours.ToString() + “ час : “ + minutes.ToString() + “ мин : “ + seconds.ToString() + “ с”;

// Восстановление пути от конечного узла до начального

while (end\_index != begin\_index) {

for (int i = 0; i < SIZE; i++) {

if (a[i][end\_index] != 0) {

int temp = weight– a[i][end\_index];

if (temp == d[i]) {

weight = temp;

end\_index = i;

ver[k] = i;

k++;

}

}

}

}

// Вывод пути в обратном порядке

for (int i = k– 1; i >= 0; i–– ) {

this– >textBox4– >Text += Convert::ToString(“ “ + (ver[i] + 1));

}

// Освобождение выделенной памяти

delete[] d;

delete[] v;

delete[] ver;

}

#pragma endregion

String^ filename = “file\_with\_Coordinates.txt”;

private: System::Void бирюзовыйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::Turquoise; //для берюзового цвета

}

private: System::Void красныйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::Red; //для красного цвета

}

private: System::Void жёлтыйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::Yellow; //для желтого цвета

}

private: System::Void белыйToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >BackColor = System::Drawing::Color::White; //для белого цвета

}

private: System::Void русскийToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >Text = “Основное окно”;

language\_now = 0;

this– >label2– >Text = “Добавить расстояние между двумя зданиями”;

this– >label3– >Text = “Введите вершину 1:”;

this– >label4– >Text = “Введите вершину 2:”;

this– >label5– >Text = “Введите расстояния между домами:”;

this– >label6– >Text = “Количество домов:”;

this– >label7– >Text = “от:”;

this– >label8– >Text = “до:”;

this– >label9– >Text = “результат:”;

this– >label10– >Text = “время затраченное на путь”;

this– >label11– >Text = “Расстояние”;

this– >button1– >Text = “Найти краткий путь”;

this– >button2– >Text = “Ввод”;

this– >бирюзовыйToolStripMenuItem– >Text = “Бирюзовый”;

this– >красныйToolStripMenuItem– >Text = “Красный”;

this– >жёлтыйToolStripMenuItem– >Text = “Жёлтый”;

this– >белыйToolStripMenuItem– >Text = “Белый”;

this– >цветОкнаToolStripMenuItem– >Text = “Цвет окна”;

this– >языкToolStripMenuItem– >Text = “Язык”;

this– >настройкиToolStripMenuItem– >Text = “Настройки”;

this– >англToolStripMenuItem– >Text = “Английский”;

this– >белорсукийToolStripMenuItem– >Text = “Белорсукий”;

this– >русскийToolStripMenuItem– >Text = “Русский”;

this– >справкаToolStripMenuItem– >Text = “Справка”;

this– >выходToolStripMenuItem– >Text = “Выход”;

}

private: System::Void англToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >Text = “Main window”;

language\_now = 1;

this– >label2– >Text = “Add distance between two buildings”;

this– >label3– >Text = “Enter vertex 1:”;

this– >label4– >Text = “Enter vertex 2:”;

this– >label5– >Text = “Enter distances between houses:”;

this– >label6– >Text = “Number of houses:”;

this– >label7– >Text = “from:”;

this– >label8– >Text = “to:”;

this– >label9– >Text = “result:”;

this– >label10– >Text = “time spent on the path”;

this– >label11– >Text = “Distance”;

this– >button1– >Text = “Find shortest path”;

this– >button2– >Text = “Input”;

this– >бирюзовыйToolStripMenuItem– >Text = “Turquoise”;

this– >красныйToolStripMenuItem– >Text = “Red”;

this– >жёлтыйToolStripMenuItem– >Text = “Yellow”;

this– >белыйToolStripMenuItem– >Text = “White”;

this– >цветОкнаToolStripMenuItem– >Text = “Window color”;

this– >языкToolStripMenuItem– >Text = “Language”;

this– >настройкиToolStripMenuItem– >Text = “Settings”;

this– >англToolStripMenuItem– >Text = “English”;

this– >белорсукийToolStripMenuItem– >Text = “Belarusian”;

this– >русскийToolStripMenuItem– >Text = “Russian”;

this– >справкаToolStripMenuItem– >Text = “help”;

this– >выходToolStripMenuItem– >Text = “Exit”;

}

private: System::Void белорссукийToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this– >Text = “Асноўнае акно”;

language\_now = 2;

this– >label2– >Text = “Дадаць адлегласць паміж двума будынкамі”;

this– >label3– >Text = “Увядзіце вершыню 1:”;

this– >label4– >Text = “Увядзіце вершыню 2:”;

this– >label5– >Text = “Увядзіце адлегласць паміж домамі:”;

this– >label6– >Text = “Колькасць домаў:”;

this– >label7– >Text = “ад:”;

this– >label8– >Text = “да:”;

this– >label9– >Text = “вынік:”;

this– >label10– >Text = “час, затрачаны на шлях”;

this– >label11– >Text = “Адлегласць”;

this– >button1– >Text = “Знайсці кароткі шлях”;

this– >button2– >Text = “Увод”;

this– >бирюзовыйToolStripMenuItem– >Text = “Бірюзавы”;

this– >красныйToolStripMenuItem– >Text = “Чырвоны”;

this– >жёлтыйToolStripMenuItem– >Text = “Жоўты”;

this– >белыйToolStripMenuItem– >Text = “Белы”;

this– >цветОкнаToolStripMenuItem– >Text = “Колер акна”;

this– >языкToolStripMenuItem– >Text = “Мова”;

this– >настройкиToolStripMenuItem– >Text = “Налады”;

this– >англToolStripMenuItem– >Text = “Англійская”;

this– >белорсукийToolStripMenuItem– >Text = “Беларуская”;

this– >русскийToolStripMenuItem– >Text = “Руская”;

this– >справкаToolStripMenuItem– >Text = “інструкцыя”;

this– >выходToolStripMenuItem– >Text = “Выхад”;

}

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e)

{

try

{

int size\_array;

// Проверка, является ли текстовое поле пустым

if (this– >textBox1– >Text == ““)

{

// Чтение размера массива из файла

StreamReader^ reader1 = gcnew StreamReader(filename);

String^ line;

if ((line = reader1– >ReadLine()) != nullptr)

{

array<String^>^ tokens = line– >Split(' ');

size\_array = tokens– >Length;

size\_array–– ;

}

reader1– >Close();

}

else

{

// Чтение размера массива из текстового поля

try

{

size\_array = Convert::ToInt32(this– >textBox1– >Text);

if (size\_array <= 1)

{

MessageBox::Show(“Invalid input! Please enter a positive number.”);

return;

}

}

catch (...)

{

}

}

bool sizeChanged = false;

int oldSizeArray = 0;

// Проверка, изменилось ли количество вершин

bool fileExists = File::Exists(filename);

if (fileExists)

{

// Чтение данных из файла

StreamReader^ reader = gcnew StreamReader(filename);

String^ line;

if ((line = reader– >ReadLine()) != nullptr)

{

array<String^>^ tokens = line– >Split(' ');

oldSizeArray = tokens– >Length;

oldSizeArray–– ;

if (size\_array != oldSizeArray)

{

sizeChanged = true;

}

}

reader– >Close();

}

// Создание нового массива для работы

int\*\* dynamicArray\_for\_work = new int\* [size\_array];

for (int i = 0; i < size\_array; i++)

{

dynamicArray\_for\_work[i] = new int[size\_array];

}

// Создание нового массива для изменений

int\*\* dynamicArray\_for\_change = new int\* [size\_array];

for (int i = 0; i < size\_array; i++)

{

dynamicArray\_for\_change[i] = new int[size\_array];

}

if (fileExists)

{

// Чтение данных из файла и заполнение массивов

StreamReader^ reader = gcnew StreamReader(filename);

String^ line;

int row = 0;

while ((line = reader– >ReadLine()) != nullptr)

{

array<String^>^ tokens = line– >Split(' ');

for (int col = 0; col < tokens– >Length– 1; col++)

{

int value;

String^ token = tokens[col];

value = Convert::ToInt32(token);

if (value >= 0)

{

if (row < size\_array && col < size\_array)

{

dynamicArray\_for\_work[row][col] = value;

dynamicArray\_for\_change[row][col] = value;

}

}

else

{

dynamicArray\_for\_work[row][col] = 0;

dynamicArray\_for\_change[row][col] = 0;

}

}

row++;

if (row == size\_array)

{

break;

}

}

reader– >Close();

}

else

{

// Обработка случая отсутствия файла

// ...

}

// Обработка изменений размера массива

if (sizeChanged)

{

if (oldSizeArray > size\_array)

{

// Копирование данных из dynamicArray\_for\_change в dynamicArray\_for\_work только для нового размера

for (int i = 0; i < size\_array; i++)

{

for (int j = 0; j < size\_array; j++)

{

dynamicArray\_for\_work[i][j] = dynamicArray\_for\_change[i][j];

}

}

}

else

{

// Копирование данных из dynamicArray\_for\_change в dynamicArray\_for\_work только для старого размера

for (int i = 0; i < oldSizeArray; i++)

{

for (int j = 0; j < oldSizeArray; j++)

{

dynamicArray\_for\_work[i][j] = dynamicArray\_for\_change[i][j];

}

}

}

}

// Обновление массива с новыми значениями из текстовых полей

String^ graphFromTo = this– >textBox1\_1– >Text;

String^ graphSourceTarget = this– >textBox2\_1– >Text;

String^ distance = this– >textBox\_distance– >Text;

// Проверка ввода

if (Convert::ToInt32(graphFromTo) <= 0 || Convert::ToInt32(graphSourceTarget) <= 0)

{

if (language\_now == 0)

{

MessageBox::Show(“Ошибка: Ввод графа осуществляется с единицы (1)”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Error: Graph input is performed from one (1)”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Памылка: Увод графа здзяйсняецца з адзінкі (1)”);

}

return;

}

if (Convert::ToInt32(graphFromTo) > size\_array || Convert::ToInt32(graphSourceTarget) > size\_array)

{

if (language\_now == 0)

{

MessageBox::Show(“Ошибка: Начало или конец графа превышает значение его размерности”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Error: Start or end of the graph exceeds its size”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Памылка: Пачатак або канец графа перавышае значэнне яго памеру”);

}

return;

}

if (graphFromTo == graphSourceTarget)

{

if (language\_now == 0)

{

MessageBox::Show(“Ввод невозможен так как вы уже находитесь в этой точке”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Input is not possible as you are already at this point”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Увод немагчымы, бо вы ўжо знаходзіцеся ў гэтай кропцы”);

}

return;

}

if (Convert::ToInt32(distance) <= 0)

{

if (language\_now == 0)

{

MessageBox::Show(“Расстояние между домами не может быть равно 0 или быть отрицательным”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Distance between houses cannot be equal to 0 or negative”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Адлегласць паміж домамі не можа быць роўнай 0 або адмоўнай”);

}

return;

}

// Обновление значений в массиве

dynamicArray\_for\_work[Convert::ToInt32(graphFromTo)– 1][Convert::ToInt32(graphSourceTarget)– 1] = Convert::ToInt32(distance);

dynamicArray\_for\_work[Convert::ToInt32(graphSourceTarget)– 1][Convert::ToInt32(graphFromTo)– 1] = Convert::ToInt32(distance);

// Запись массива обратно в файл

StreamWriter^ sw = gcnew StreamWriter(filename);

for (int i = 0; i < size\_array; i++)

{

for (int j = 0; j < size\_array; j++)

{

if (dynamicArray\_for\_work[i][j] <= 0)

{

sw– >Write(“0 “);

}

else

{

sw– >Write(dynamicArray\_for\_work[i][j] + “ “);

}

}

sw– >WriteLine();

}

sw– >Close();

// Отрисовка графа на панели

Panel^ graphPanel = this– >panel1;

Graphics^ g = graphPanel– >CreateGraphics();

g– >Clear(Color::White); // Очистка области рисования перед отрисовкой

DrawGraph(g, dynamicArray\_for\_work, size\_array);

// Освобождение памяти

for (int i = 0; i < size\_array; i++)

{

delete[] dynamicArray\_for\_work[i];

}

delete[] dynamicArray\_for\_work;

for (int i = 0; i < size\_array; i++)

{

delete[] dynamicArray\_for\_change[i];

}

delete[] dynamicArray\_for\_change;

}

catch (...)

{

// Обработка ошибок при вводе в текстовое поле

if (language\_now == 0)

{

MessageBox::Show(“Ошибка при вводе в текстовое поле”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Error when entering into textbox”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Памылка пры ўводзе ў тэкставае поле”);

}

}

}

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e)

{

try

{

// Чтение размерности графа из файла

StreamReader^ reader1 = gcnew StreamReader(filename);

String^ line;

int size\_array;

if ((line = reader1– >ReadLine()) != nullptr)

{

array<String^>^ tokens = line– >Split(' ');

size\_array = tokens– >Length;

size\_array–– ;

}

reader1– >Close();

// Получение начальной и конечной точек из текстовых полей

int begin\_index, end\_index;

begin\_index = Convert::ToInt32(this– >textBox2– >Text);

end\_index = Convert::ToInt32(this– >textBox3– >Text);

// Проверка наличия точек в графе

if (begin\_index == end\_index)

{

if (language\_now == 0)

{

MessageBox::Show(“Вы уже находитесь в этой точке”);

}

else if (language\_now == 1)

{

MessageBox::Show(“You are already at this point”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Вы ўжо знаходзіцеся ў гэтай кропцы”);

}

this– >textBox4– >Clear();

this– >textBox5– >Clear();

this– >textBox6– >Clear();

return;

}

// Проверка ввода начальной и конечной точек

if (begin\_index <= 0 || end\_index <= 0)

{

if (language\_now == 0)

{

MessageBox::Show(“Ошибка: Начало или конец графа не может быть равен 0 или отрицательному числу”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Error: Start or end of the graph cannot be equal to or less than 0”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Памылка: Пачатак або канец графа не можа быць роўны 0 або адмоўным лікам”);

}

this– >textBox4– >Clear();

this– >textBox5– >Clear();

this– >textBox6– >Clear();

return;

}

// Проверка наличия начальной и конечной точек в графе

if (begin\_index > size\_array || end\_index > size\_array)

{

if (language\_now == 0)

{

MessageBox::Show(“Ошибка: Начало или конец графа превышает значение его размерности”);

}

else if (language\_now == 1)

{

MessageBox::Show(“Error: Start or end of the graph exceeds its size”);

}

else if (language\_now == 2)

{

MessageBox::Show(“Памылка: Пачатак або канец графа перавышае значэнне яго памеру”);

}

this– >textBox4– >Clear();

this– >textBox5– >Clear();

this– >textBox6– >Clear();

return;

}

// Создание двумерного массива для хранения графа

int\*\* dynamicArray = new int\* [size\_array];

for (int i = 0; i < size\_array; i++)

{

dynamicArray[i] = new int[size\_array];

}

// Чтение графа из файла и заполнение массива

StreamReader^ reader2 = gcnew StreamReader(filename);

int row = 0;

while ((line = reader2– >ReadLine()) != nullptr)

{

array<String^>^ tokens = line– >Split(' ');

for (int col = 0; col < tokens– >Length– 1; col++)

{

int value;

String^ token = tokens[col];

value = Convert::ToInt32(token);

if (value >= 0)

{

if (row < size\_array && col < size\_array)

{

dynamicArray[row][col] = value;

}

}

else

{

dynamicArray[row][col] = 0;

}

}

row++;

if (row == size\_array)

{

break;

}

}

reader2– >Close();

// Выполнение алгоритма Дейкстры

dijkstra(dynamicArray, size\_array, begin\_index– 1, end\_index– 1);

// Освобождение памяти, выделенной для массива

for (int i = 0; i < size\_array; i++)

{

delete[] dynamicArray[i];

}

delete[] dynamicArray;

}

catch (...)

{

// Обработка ошибок

}

}

private: System::Void panel1\_Paint(System::Object^ sender, System::Windows::Forms::PaintEventArgs^ e)

{

int size\_array = 0;

bool sizeChanged = false;

int oldSizeArray = 0;

// Чтение размерности графа из файла

StreamReader^ reader = gcnew StreamReader(filename);

String^ line;

if ((line = reader– >ReadLine()) != nullptr)

{

array<String^>^ tokens = line– >Split(' ');

oldSizeArray = tokens– >Length;

oldSizeArray–– ;

if (size\_array != oldSizeArray)

{

sizeChanged = true;

size\_array = oldSizeArray; // Присваивание значения переменной size\_array

}

}

reader– >Close();

// Создание нового массива для работы

int\*\* dynamicArray\_for\_work = new int\* [size\_array];

for (int i = 0; i < size\_array; i++)

{

dynamicArray\_for\_work[i] = new int[size\_array];

}

// Чтение данных из файла и заполнение массива

reader = gcnew StreamReader(filename); // Переиспользуем переменную reader

int row = 0;

while ((line = reader– >ReadLine()) != nullptr)

{

array<String^>^ tokens = line– >Split(' ');

for (int col = 0; col < tokens– >Length– 1; col++) // Изменено условие цикла

{

int value;

String^ token = tokens[col];

value = Convert::ToInt32(token);

if (value >= 0)

{

if (row < size\_array && col < size\_array)

{

dynamicArray\_for\_work[row][col] = value;

}

}

else

{

dynamicArray\_for\_work[row][col] = 0;

}

}

row++;

if (row == size\_array)

{

break;

}

}

reader– >Close();

// Получение объекта Graphics для панели

Graphics^ g = panel1– >CreateGraphics();

// Вызов функции отрисовки графа

DrawGraph(g, dynamicArray\_for\_work, size\_array);

// Освобождение памяти, выделенной для массива

for (int i = 0; i < size\_array; i++)

{

delete[] dynamicArray\_for\_work[i];

}

delete[] dynamicArray\_for\_work;

}

private: System::Void справкаToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

// Открытие файла справки

system((“start “ + std::string(“Справка.chm”)).c\_str());

}

private: System::Void выходToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

exit(0);

}

};

}

# ПРИЛОЖЕНИЕ Б (справочное) Результаты работы программы

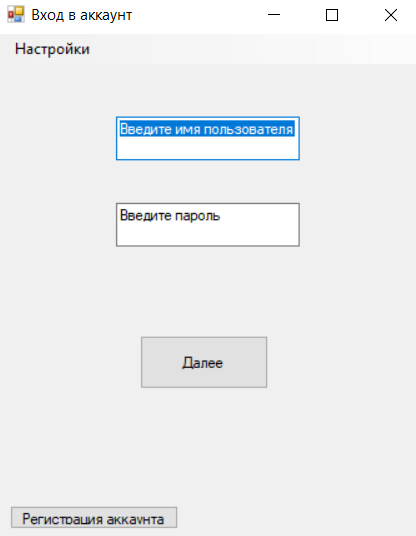


Рисунок Б. – Результат функции «Запустить приложение»

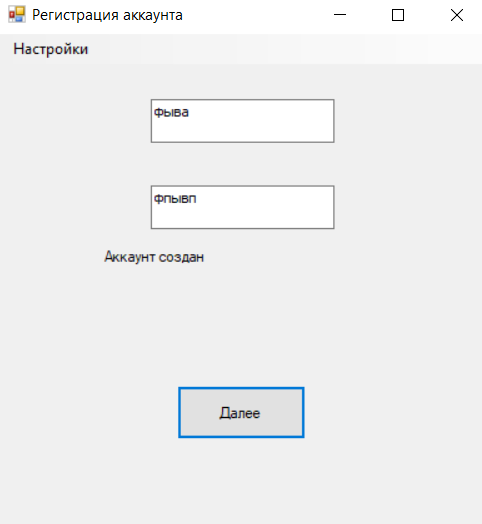


Рисунок Б. – Результат успешной регистрации

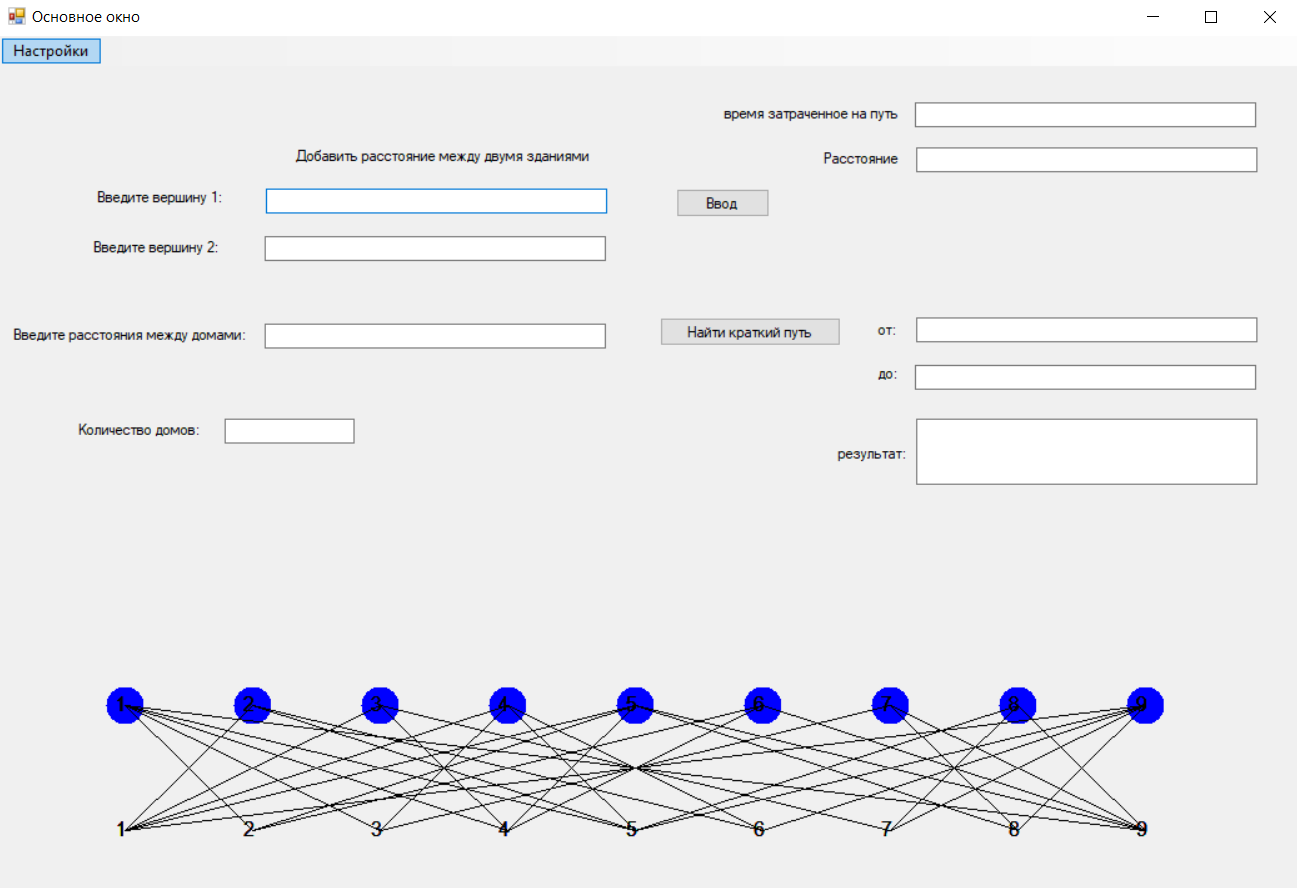


Рисунок Б. – Результат успешного входа в аккаунт

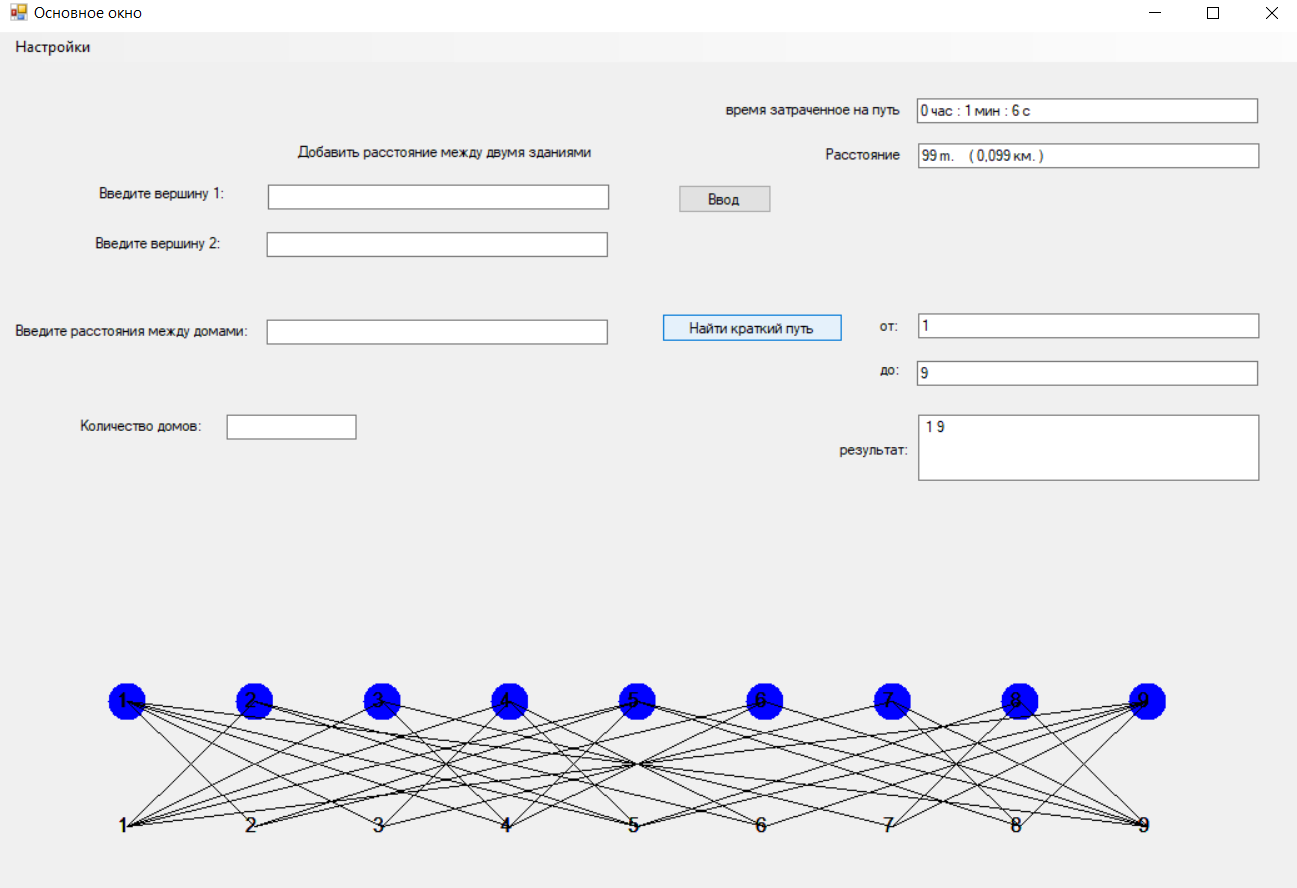


Рисунок Б. – Результат успешного поиска пути

# 

КБП

У

Т. Контр.

Разраб.

Неворский И.А.

Провер.

Шаляпин Ю.В.

Реценз.

Утверд.

Н. Контр.

№ Докум.

Подпись

Дат

Изм.

Лист

Масса

Лит.

Масштаб

Лист 1

Листов 2

Инв.№подл.

Подп. и дата

Взам.инв.№в.

Инв.№ дубл.

Подп. и дата

КП Т.197018.404 ГЧ

КП Т.197018.404 ГЧ



*Программа для нахождения наименьшего пути между зданиями в городе методом расчета минимального пути между вершинами графа по алгоритму Дейкстры и визуализация полученных результатов и алгоритма*

Блок-схема входа в систему

КБП

У

Т. Контр.

Разраб.

Неворский И.А.

Провер.

Шаляпин Ю.В.

Реценз.

Утверд.

Н. Контр.

№ Докум.

Подпись

Дат

Изм.

Лист

Масса

Лит.

Масштаб

Лист 2

Листов 2

Инв.№подл.

Подп. и дата

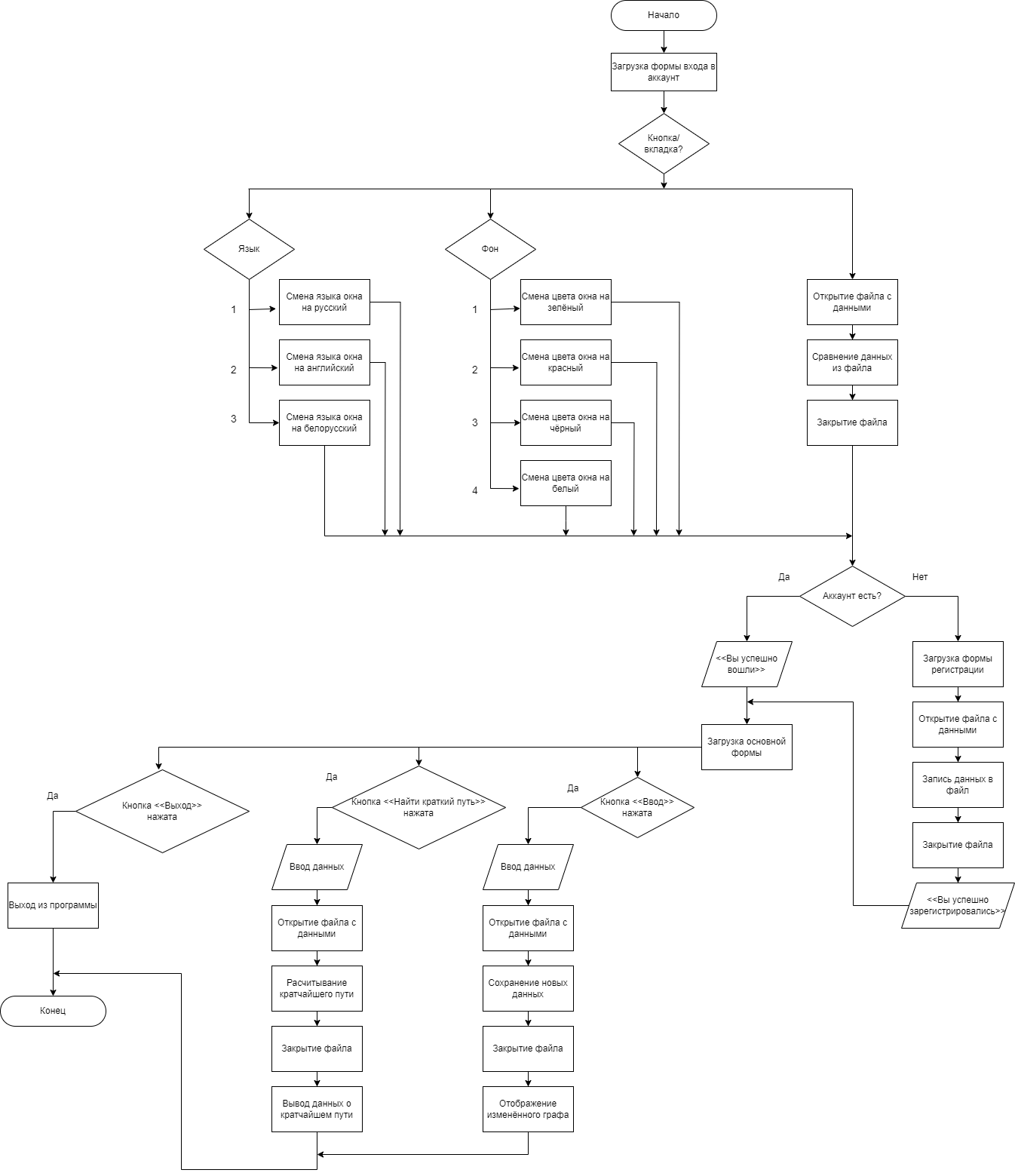
Взам.инв.№в.

Инв.№ дубл.

Подп. и дата

КП Т.197018.401 ГЧ

КП Т.197018.404 ГЧ



*Программа для нахождения наименьшего пути между зданиями в городе методом расчета минимального пути между вершинами графа по алгоритму Дейкстры и визуализация полученных результатов и алгоритма*

Блок-схема работы программы (общая)

**Удостоверяющий лист**

электронного документа– курсовой проект

Тема КП «Программа для нахождения наименьшего пути между зданиями в городе методом расчета минимального пути между вершинами графа по алгоритму Дейкстры и визуализация полученных результатов и алгоритма»

Обозначение КП Т.197018.404 .

Разработчик Неворский И. А. Руководитель Шаляпин Ю.В.

(Ф.И.О.) (Ф.И.О.)

Подписи лиц, ответственных за разработку электронного документа

|  |  |  |
| --- | --- | --- |
| Состав электронногодокумента | Разработчик | Руководитель |
| Пояснительная записка (на бумажном носителе формата А4), Неворский\_ПЗ.docх |  |  |
| ГЧ, Блок– схема работы программы (общая).docx |  |  |
| ГЧ, Блок– схема входа в систему.docx |  |  |
| Папка с проектом «kursov» |  |  |
| Установочный пакет программного средства «kursov.exe» |  |  |
| Тип носителя: Оптический диск |  |  |

**Этикетка**

для курсового проекта

**Курсовой проект**

Тема «Программа для нахождения наименьшего пути между зданиями в городе методом расчета минимального пути между вершинами графа по алгоритму Дейкстры и визуализация полученных результатов и алгоритманахождения кратчайшего пути»

КП Т.197018.404

Разработан

Утвержден

Разработчик Неворский И. А.

Руководитель: Шаляпин Ю.В.

Технические средства: Компьтер с процессором от 1 ГГц, ОЗУ от 2 ГБ, 5 ГБ свободного места на жестком диске, монитор, мышь, клавиатура.

Программные средства: Операционная система MS Windows 10 Pro, IDE Microsoft Visual Studio 2022, Windows Kits, Windows Form, MS Visio 2007, текстовые редакторы MS Office Word 2016, Notepad.exe.

**Состав документа:**

Пояснительная записка– Неворский\_ПЗ.docх

Графическая часть– Блок– схема работы программы (общая).docx, Блок– схема входа в систему.docx.

Папка с проектом– kursov

Установочный пакет программного средства– kursov.exe

Сведения о защите информации: пароль отсутствует