ЧАСТНОЕ УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«КОЛЛЕДЖ БИЗНЕСА И ПРАВА»

БРЕСТСКИЙ ФИЛИАЛ

АВТОМАТИЗИРОВАННОЕ РАБОЧЕЕ МЕСТО

«НАЛОГОВОГО ИНСПЕКТОРА»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ ПО ДИСЦИПЛИНЕ

«ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ»

КП Т.091021.401

Руководитель проекта (Ю.В. Куцун)

Учащийся (А.Е. Чернов)

Отметка за работу \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Работа защищена на оценку\_\_\_\_\_\_\_\_

Преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2022

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3

КП Т.091021.401

Разраб.

А.Е. Чернов

Провер.

Ю. В. Куцун

Т. контр.

Н. контр.

Утверд.

Автоматизированное рабочее место налогового инспектора

Пояснительная записка

Лит.

Листов

34

КБиП

Содержание

[Введение 4](#_Toc106443114)

[1 Объектно-ориентированный анализ и проектирование системы 5](#_Toc106443115)

[1.1 Сущность задачи 5](#_Toc106443116)

[1.2 Проектирование модели 5](#_Toc106443117)

[2 Вычислительная система 6](#_Toc106443118)

[2.1 Требования к аппаратным и операционным ресурсам 6](#_Toc106443119)

[2.2 Инструменты разработки 6](#_Toc106443120)

[3 Проектирование задачи 8](#_Toc106443121)

[3.1 Требования к приложению 8](#_Toc106443122)

[3.2 Концептуальный прототип 8](#_Toc106443123)

[3.3 Функции: логическая и физическая организация 9](#_Toc106443124)

[3.4 Функции и элементы управления 9](#_Toc106443125)

[4 Описание программного средства 10](#_Toc106443126)

[4.1 Общие сведения 10](#_Toc106443127)

[4.2 Функциональное назначение 10](#_Toc106443128)

[4.3 Входные данные 10](#_Toc106443129)

[4.4 Выходные данные 10](#_Toc106443130)

[5 Методика испытаний 11](#_Toc106443131)

[5.1 Технические требования 11](#_Toc106443132)

[5.2 Порядок проведения испытаний 11](#_Toc106443133)

[5.2.1 Функциональное тестирование 11](#_Toc106443134)

[5.2.2 Полное тестирование 12](#_Toc106443135)

[6 Применение 19](#_Toc106443136)

[6.1 Назначение программы 19](#_Toc106443137)

[6.2 Условия применения 19](#_Toc106443138)

[Заключение 19](#_Toc106443139)

[Список использованных источников 21](#_Toc106443140)

[Приложение А 22](#_Toc106443141)

# Введение

Компьютерные системы – комплекс компьютерного, активного сетевого оборудования, системного и офисного программного обеспечения, необходимого для успешного функционирования комплексной автоматизированной системы учёта уплаты налогов. Системное и офисное программное обеспечение, программно-аппаратные комплексы защиты информации.

Заданием курсового проекта стала разработка программы «Автоматизированное рабочее место налогового инспектора».

Объект автоматизации – учёт, редактирование, обработка и предоставление информации о лицах, которых нужно проверить, из базы данных.

В данной программе присутствуют такие функции как:

* просмотр информации о лицах непрошедших проверку;
* добавление, изменение и удаление записей;
* поиск записей и просмотр подходящих записей;
* чтение и запись информации из файла, а также выбор редактируемого файла;

Актуальность проекта состоит в том, чтобы, используя информационные технологии, создать приложение, способное упростить работу налогового инспектора с лицами, непрошедшими проверку.

В результате работы над курсовым проектом было разработано приложение и составлена пояснительная записка.

Пояснительная записка к курсовому проекту «Автоматизированное рабочее место налогового инспектора», включает в себя шесть разделов, содержащих необходимую и достаточную информацию по использованию данного приложения, подробное описание программы и вычислительной системы.

Первый раздел «Объектно-ориентированный анализ и проектирование системы» рассматривает организационную сущность задачи, описываются входные и выходные данные, описывает назначение курсового проекта.

Второй раздел «Вычислительная система» имеет описание вычислительной системы, а именно: технические характеристики персонального компьютера, требования, предъявляемые к персональному компьютеру, описание операционной системы, выбранных языков реализации, среды разработки.

Третий раздел «Проектирование задачи» раскрывает сущность задачи в ключевых абстракциях предметной области и описывает интерфейс классов.

Четвертый раздел «Описание программного средства» отражает общее представление о программе.

В пятом разделе «Методика испытаний» рассматривается методика испытаний программы и процесс отладки.

Шестой раздел «Применение» содержит информацию, необходимую в процессе эксплуатации программного средства. Обозначены требуемые характеристики к компьютеру, условия применения и назначение программы.

В разделе «Заключение» подводятся итоги и описываются преимущества разработанного программного средства.

В «Список использованных источников» приводится список использованной литературы и ссылок интернета.

В приложение включен текст программы.

# 1 Объектно-ориентированный анализ и проектирование системы

## 1.1 Сущность задачи

Автоматизированный контроль за уплатой налогов поможет инспектору контролировать свою работу и управлять временем и управлять записями. Это повысит продуктивность и уменьшит количество используемой бумаги.

Целью курсовой работы является учёт и дальнейшее использование списка лиц, которых следует проверить, их имя, фамилия,

Подробное изучение этих вопросов позволит выявить конкретные методы оценки, учета и контроля материальных ресурсов, используемых организацией, а также предложить пути улучшения использования материалов в рамках возможных способов ведения учёта.

Программа «Автоматизированное рабочее место налогового инспектора» красива, функциональна, понятна и удобна в использовании. Главным достоинством программы является возможность вести учёт материалов и хранить информацию о уже имеющихся записях без подключения к сети Интернет.

Разрабатываемая программа имеет больший потенциал:

– может работать без подключения к Интернет;

– имеет структурированную форму учёта лиц, для проверки;

– возможность сохранения информации в документ формата .txt;

– для работы с программой не надо никого обучать, так как она нативна и понятна,

а также содержит руководство пользования;

Задачей автоматизации являлась быстрая подача информации, удобство в редактировании и сохранение документом.

Контингентом пользователей приложения могут быть любые люди без ограничения по возрасту, полу, физическим и психическим особенностям.

## 1.2 Проектирование модели

Целью проектирования модели являлось преобразование общих нечетких знаний о предназначении будущего программного обеспечения и сравнительно точные требования к нему.

# 2 Вычислительная система

## 2.1 Требования к аппаратным и операционным ресурсам

Разрабатываемый программный продукт «Автоматизированное рабочее место налогового инспектора», предназначена для работы в операционной системе Windows10 и Windows 11.

Для нормального функционирования данной программы необходим компьютер со следующими характеристиками:

* процессор AMD Ryzen 3 1200 или совместимый Intel процессор;
* минимальный объём оперативной памяти – 4 Гбайт;
* жёсткий диск либо твердотелый накопитель с минимальным объёмом 256 Гбайт;
* монитор типа HD+ и выше;
* клавиатура, мышь.

Требуется программное обеспечение:

– наличие NET.Framework 4.8;

При использовании более слабых конфигураций возможны задержки при работе программы, либо неработоспособность некоторых функций.

## 2.2 Инструменты разработки

Для разработки курсового проекта использовалось следующее программное обеспечение:

* Операционная система Microsoft Windows 11 Pro;
* Microsoft Visual Studio 2022;
* Microsoft Office 365;

Операционная система — это программа, которая загружается при включении компьютера. Она производит диалог с пользователем, осуществляет управление компьютером, его ресурсами, запускает другие (прикладные) программы на выполнение.

На сегодняшний момент операционная система Windows фирмы Microsoft во всех ее проявлениях является самой распространенной операционной системой на ПК: в мире более ста пятидесяти миллионов IBM PC-совместимых компьютеров, и система Windows установлена на ста миллионах из них. Операционные системы, развиваясь вместе с ЭВМ, прошли длинный путь от простейших программ в машинных кодах длинной в несколько килобайт до монстров, написанных на языках высокого уровня, размер которых исчисляется десятками мегабайт. Такой значительный рост размера операционных систем обусловлен, главным образом, стремлением разработчиков «украсить» операционную систему, расширить ее возможности, добавить возможности, изначально несвойственные операционным системам, а также сделать интерфейс пользователя интуитивным. Все эти попытки дали свои результаты, и положительные, и отрицательные.

Microsoft **Windows 11** – это не только новая операционная система от Microsoft, это – совершенно новая оболочка для офиса и дома, для развлечений и учёбы, для отдыха и разработки. Недаром многие разработчики говорят о том, что она станет «революционной» ОС, которая изменит представления о взаимодействии с компьютером. Система призвана стать единой для разных устройств, таких как [персональные компьютеры](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%81%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80), [планшеты](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%BD%D1%88%D0%B5%D1%82%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80) и другие X64 и ARM-совместимые устройства.

С самого начала разработчики начали позиционировать систему, как полностью новую в визуальном плане, но с многочисленными улучшениями. Учитывались преимущества Windows 10, а также запросы со стороны пользователей.

Примерно на протяжении более полугода до выхода можно было принять участие в альфа и бета-тестировании и рассказать о возможностях улучшения или доработок.

# 3 Проектирование задачи

## 3.1 Требования к приложению

При разработке требований к системе на этапе формирования требований, прежде всего, необходимо было определить область действия разрабатываемой системы и получить точное представление о ее желаемых возможностях.

Созданное пользовательское меню и интерфейс должны обеспечивать доступ ко всем основным функциям программы. Во всех ветвях программы одинаковые действия должны инициализироваться одинаковыми клавишами.

Источниками данных для создания модели системных прецедентов являлись разработанные на предыдущем этапе бизнес-модели.

Программное обеспечение должно соответствовать следующим требованиям:

* быть удобным в использовании;
* иметь лёгкий для освоения интерфейс, который не должен создавать перед пользователем преграду или неудобства в работе с программой.

Должны быть реализованы простые способы управления, которые будут уместны при работе с интерактивным приложением. Должны быть использованы такие объекты управления как кнопки, поля для ввода и другие объекты, которые будут уместны в том или ином случае.

Элементы управления должны быть расположены в оптимальном порядке. Рационально расположенные элементы управления позволят наиболее быстро и корректно заносить информацию в базу данных, а также выполнять над ними операции.

Созданное пользовательское меню должно обеспечивать доступ ко всем основным функциям программы.

Если данные не являются тем, что система считает допустимым, то она всё равно должна их принять, информировать об этом пользователя и обработать их соответствующим образом.

Выходные данные должны выдаваться в требуемой форме, и обязательно прокомментированными.

## 3.2 Концептуальный прототип

На концептуальном уровне управления программным средством определялось:

* цели и задача проекта;
* рассматривались альтернативные варианты для достижения намеченных результатов с оценкой отрицательных и положительных аспектов каждого варианта;
* определялись концептуальные направления реализации проекта, в частности описание предметной области, укрупнённой структуры работ и логики их выполнения;
* предварительно оценивались продолжительность проекта, а также потребности в ресурсах.

На основе информации, выявленной на этапах предварительного моделирования, выполнялась разработка концептуальной модели данных, которая использовалась в разрабатываемой системе.

Этот этап завершал процедуру предварительного моделирования и позволил представить единым форматом ту информацию, которая необходима для создания системы.

Разработанные диаграммы стали отправной точкой в процессах проектирования базы данных.

Исходя из общих требований к программному приложению, а также, учитывая область его применения и уровень пользователя, формировался концептуальный прототип программного приложения.

Под концептуальным прототипом следует понимать описание внешнего пользовательского интерфейса: системы меню и закрепленными действиями за каждым пунктом меню, диалоговых окнах и элементах управления.

Целью создания эргономичного пользовательского интерфейса являлось эффективное отображение информации для человеческого восприятия и структурированного отображения на дисплее таким образом, чтобы привлечь внимание к объектам приложения.

В приложении предусмотрено главное меню, обеспечивающие быстрый доступ к основным функциям программы.

## 3.3 Функции: логическая и физическая организация

В главном окне программы расположены следующие кнопки:

* «Отображение данных» – показывает список записей, а также удаления и поиска записей;
* «Редактировать строку» — это аналогичное с кнопкой «Отображение данных», только сс возможность ввести данные и редактировать определенную строку;
* «Добавить строку» — это очень похожее на кнопку «Редактировать строку», только после ввода данных, они записываются в конец файла;
* «Выбрать файл» — открывает окно, где присутствует возможность выбрать файл формата .txt с уже готовым списком товаров;
* «Информационное меню» — данное меню не является кнопкой, но оно сопровождает пользователя во всех представленных меню, а также имеет два дополнительных пункта. Информация — показывает информацию о программе в отдельном окне. Выход — позволяет выйти из программы.

Для представления данных использовался txt-файл.

Данные оформлены в табличный список в txt-файле.

## 3.4 Функции и элементы управления

Для доступа к функциям программы в основном используются такие элементы управления как кнопки, поля для ввода.

Для приложения был разработан адаптивный дизайн пользовательского интерфейса. Открытый и функциональный он позволяет даже не подготовленному пользователю быстро сориентироваться и начать работу с программой.

В приложении реализованы такие функции как:

* добавление записей в раздел «Отображение записей»;
* просмотр подробной информации о имеющихся товарах;
* ручное добавление, изменение и удаление записей в раздел «Отображение»;
* чтение программой txt-файлов и просмотр пользователем готового списка записей;

# 4 Описание программного средства

## 4.1 Общие сведения

Обмен информации между пользователем и компьютером осуществляется за счёт управляющих сигналов. Входные сообщения генерируются человеком с помощью средств ввода (клавиатуры, "мыши") и так далее. Программа запускается из установочного файла программы Проект.exe.

Доступ в программу осуществляется без какой-либо предварительной авторизации пользователя.

## 4.2 Функциональное назначение

Компьютерная система создана с целью упрощения работы налогового инспектора. Так же она будет удобна любому менеджеру для ведения и учета материальных ценностей, а также для ведения учета сотрудников организации.

В программе предусматриваются следующие функции и возможности:

* добавление, изменение и удаление записей;
* просмотр имеющихся записей;
* чтение и просмотр готового txt-файла со списком товаров;

## 4.3 Входные данные

Выбор действия осуществляется через основное меню программы.

Каждая кнопка отвечает за определенное действие, которое можно использовать по своему усмотрению. Кнопки представлены в программе компактно и в неизменяемом размере и виде, то есть таким образом, что даже не опытный пользователь легко найдет необходимые ему действия.

## 4.4 Выходные данные

Выходные данные представляются в виде строчной таблицы, выводящейся на экран.

# 5 Методика испытаний

## 5.1 Технические требования

Пользователь сам выполняет все функциональные обязанности по приобретению информации.

Для проведения испытаний правильности работоспособности программы использовался ноутбук со следующими характеристиками:

* процессор AMD Ryzen 5 5600H 3,3 ГГц;
* объем оперативной памяти – 16 Гбайт;
* SSD накопитель формата NWMe 512 Гбайт.

Для проведения испытаний правильности работоспособности программы необходимо также установленное программное обеспечение:

* наличие NET.Framework 4.8.
* Microsoft Visual Studio 2022.

## 5.2 Порядок проведения испытаний

### 5.2.1 Функциональное тестирование

Тестирование программного обеспечения – попытка определить, выполняет ли программа то, что от неё ожидают. Тестирование программного обеспечения – процесс выявления ошибок в программном обеспечении.

В качестве тестирования было выбрано функциональное, то есть проверка выполняемых функций по заданным критериям.

Существующие на сегодняшний день методы тестирования программного обеспечения не позволяют однозначно и полностью устранить все дефекты и ошибки и установить корректность функционирования анализируемой программы, особенно в закрытых частных программах. Поэтому все существующие методы тестирования действуют в рамках формального процесса проверки исследуемого или разрабатываемого программного обеспечения.

Такой процесс формальной проверки или верификации может доказать, что дефекты отсутствуют, с точки зрения используемого метода. То есть, нет никакой возможности точно установить или гарантировать отсутствие дефектов в программном продукте с учётом человеческого фактора, присутствующего на всех этапах жизненного цикла программного обеспечения.

Существует множество подходов к решению задачи тестирования и верификации программного обеспечения, но эффективное тестирование сложных программных продуктов – это процесс в высшей степени творческий, не сводящийся к следованию строгим и чётким процедурам или созданию таковых.

Как правило, никакое тестирование не может дать абсолютной гарантии работоспособности программы в будущем.

Для тестирования был выбран метод функционального тестирования под названием «черный ящик».

Объектом испытаний являлся программный продукт «Автоматизированное рабочее место менеджера магазина электронной техники».

Таблица 5.1 – Документирование функционального тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| Операция | Получение результата | Результат по ТЗ | Примечание |
| Запуск программы | + | + | выполнено |
| Авторизация | + | + | выполнено |
| Нажатие на кнопку «Отображение данных» | + | + | выполнено |
| Нажатие на кнопку «Редактировать строку» | + | + | выполнено |
| Нажатие на кнопку «Добавить строку» | + | + | выполнено |
| Нажатие на кнопку «Выбрать файл» | + | + | выполнено |
| Нажатие на кнопку меню «Выбрать файл» | + | + | выполнено |
| Нажатие на кнопку меню «Отображение данных» | + | + | выполнено | |
| Нажатие на кнопку меню «Добавить строку» | + | + | выполнено | |
| Нажатие на кнопку меню «Редактировать строку» | + | + | выполнено | |
| Нажатие на кнопку меню «Информация» | + | + | выполнено | |
| Нажатие на кнопку меню «Выход» | + | + | выполнено | |

Таким образом, верификация проекта была пройдена без каких-либо ошибок и показала соответствие проекта полученному заданию.

### 5.2.2 Полное тестирование

Данный этап испытаний включает последовательность следующих тестов:

* Тест 1 – проверка запуска программы;
* Тест 2 – Работоспособность кнопки «Отображение данных».
* Тест 3 – Работоспособность кнопки «Выбор файла».
* Тест 4 – Работоспособность кнопки «Редактирование строк»
* Тест 5 – Работоспособность кнопки «Добавление строк»
* Тест 6 – Работоспособность кнопки «Информация»
* Тест 7 – Работоспособность кнопки «Выход»
* Тест 8 – Работоспособность кнопки «Удалить»
* Тест 9 – Работоспособность кнопки «Редактировать строку»
* Тест 10 – Работоспособность кнопки «Добавить строку»
* Тест 11 – Работоспособность кнопки «Поиск»

Описание тестов приведено ниже.

Для начала работы необходимо запустить файл Проект.exe.

Тест 1 – При запуске программы запускается окно.

Описание теста: данный тест предназначен для проверки правильности запуска программы.

Результат работы при правильном запуске программы представлен на рисунке 5.1.

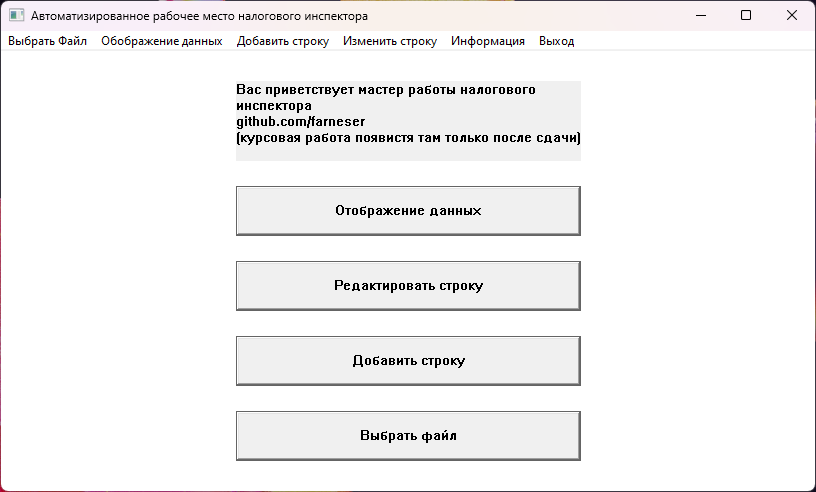


Рисунок 5.1 – Запуск программы

Тест 2 – Работоспособность кнопки «Отображение данных».

Описание теста: Данный тест предназначен для проверки открытия одного из пунктов главного меню программы, однако необходимо нажать одну из кнопок выбора файла для просмотра информации.

Результат работы программы представлен на рисунке 5.2.

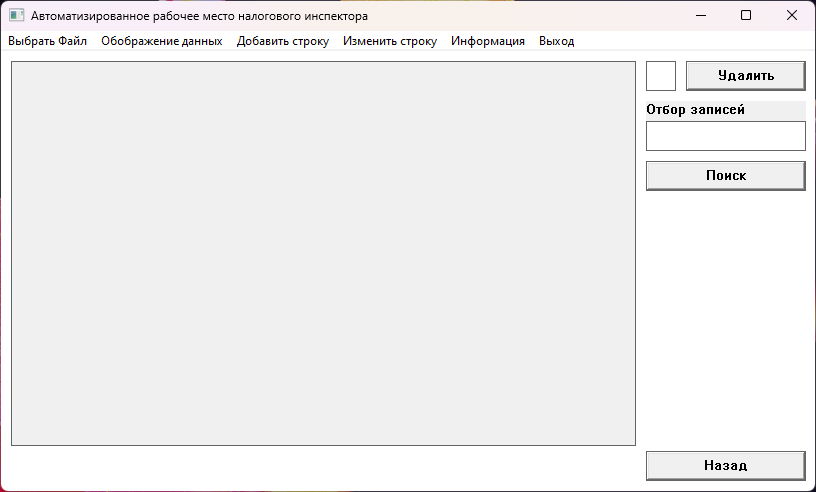


Рисунок 5.2 – Открытие меню «Отображение данных»

Тест 3 – Работоспособность кнопки «Выбор файла».

Описание теста: данный тест предназначен для проверки правильности отображения окна выбора файла. В данном меню выбрали файл kurs.txt и нажали кнопку открыть

Результат работы программы представлен на рисунке 5.3.

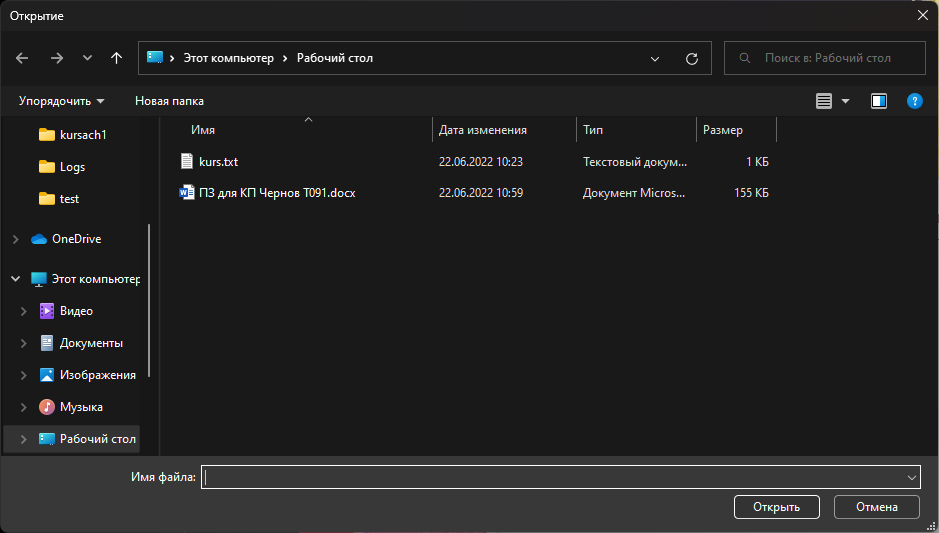


Рисунок 5.3 – Открытие меню «Выбор файла»

Тест 4 – Работоспособность кнопки «Редактирование строк»

Описание теста: Данный тест предназначен для проверки открытия пункта главного меню программы «Редактирование строк».

Результат работы программы представлен на рисунке 5.4.

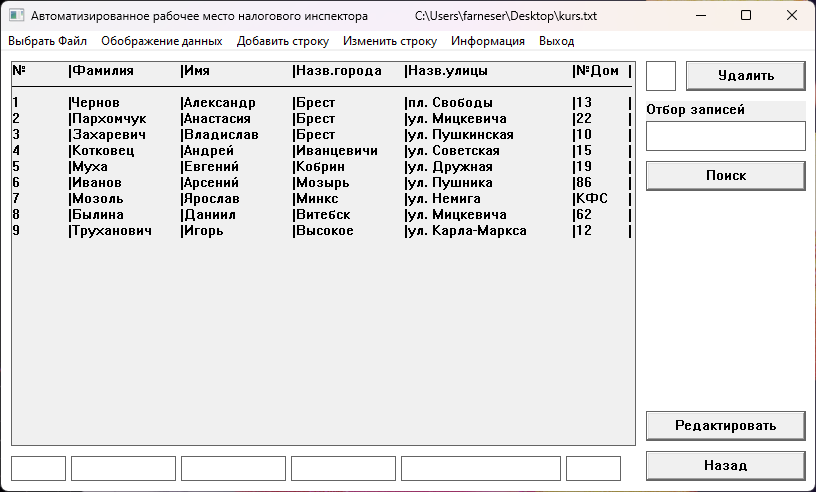


Рисунок 5.4 – Открытие меню «Редактировать строку»

Тест 5 – Работоспособность кнопки «Добавление строк»

Описание теста: Данный тест предназначен для проверки открытия пункта главного меню программы «Добавление строк».

Результат работы программы представлен на рисунке 5.5.

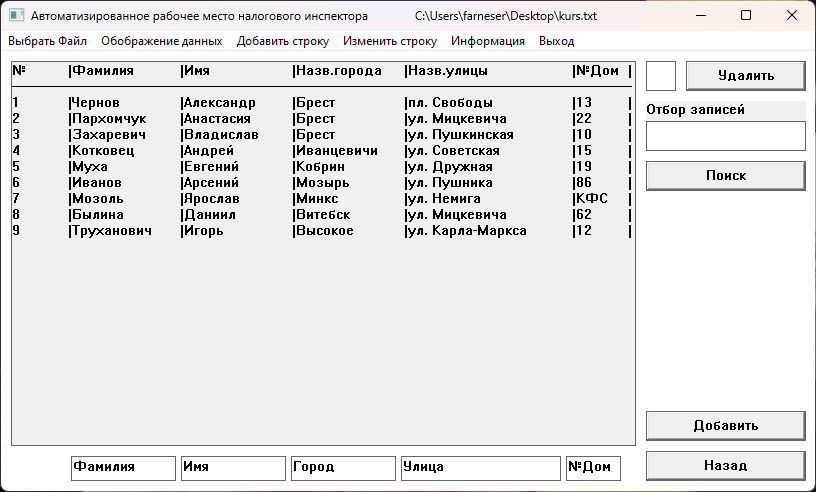


Рисунок 5.5 – Открытие меню «Добавить строку»

Тест 6 – Работоспособность кнопки «Информация»

Описание теста: Данный тест предназначен для проверки открытия пункта главного меню программы «Информация».

Результат работы программы представлен на рисунке 5.6.

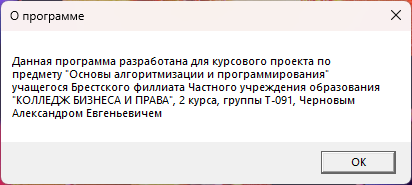


Рисунок 5.6 – Открытие меню «Информация»

Тест 7 – Работоспособность кнопки «Выход»

Описание теста: Данный тест предназначен для проверки выхода из программы.

Результат работы при нажатии кнопки «Выход» - программа завершает работу.

Тест 8 – Работоспособность кнопки «Удалить»

Описание теста: Данный тест предназначен для проверки работоспособности удаления строки.

Результат работы программы представлен на рисунке 5.7.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 5.7 – Удаление строки

Тест 9 – Работоспособность кнопки «Редактировать строку»

Описание теста: Данный тест предназначен для проверки работоспособности редактирования строки.

Результат работы программы представлен на рисунке 5.8.

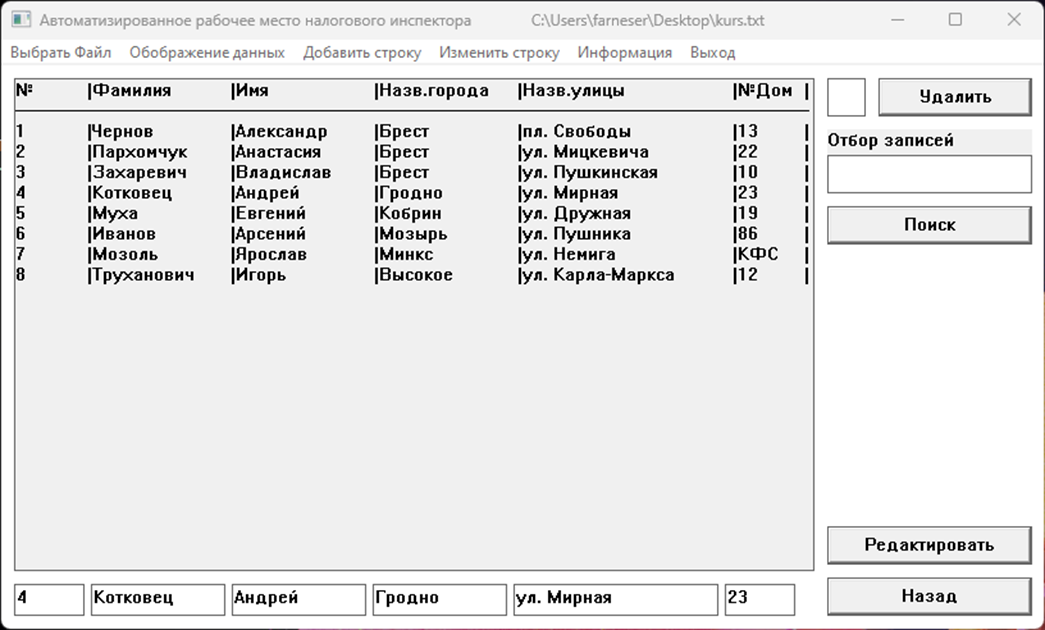


Рисунок 5.8 – Редактирование строки

Тест 10 – Работоспособность кнопки «Добавить строку»

Описание теста: Данный тест предназначен для проверки работоспособности добавления строки.

Результат работы программы представлен на рисунке 5.9.

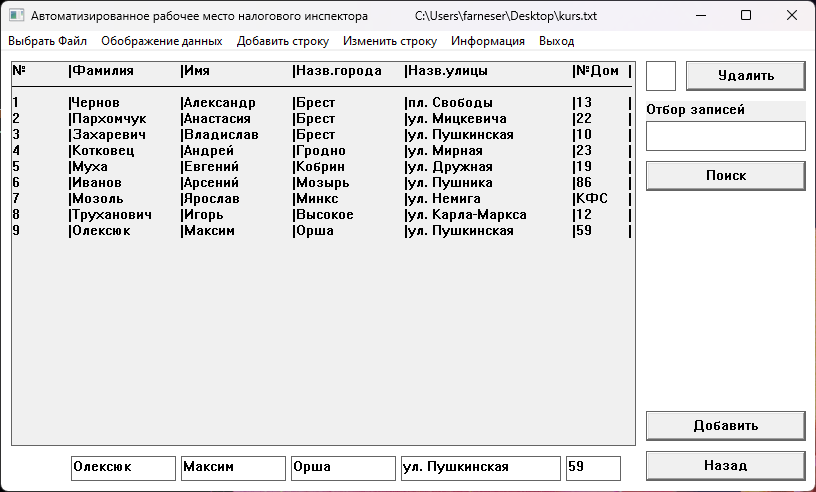


Рисунок 5.9 –Добавление строки

Тест 11 – Работоспособность кнопки «Поиск»

Описание теста: Данный тест предназначен для проверки работоспособности поиска строк.

Результат работы программы представлен на рисунке 5.10.

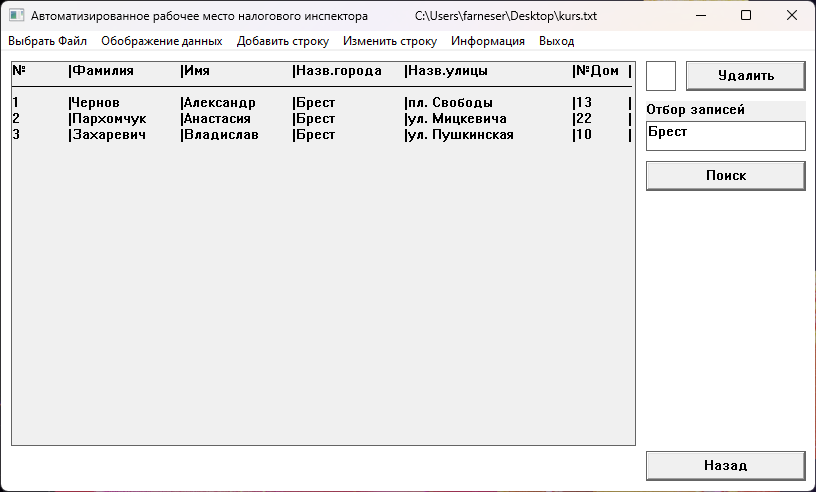


Рисунок 5.10 – Поиск строки

Таким образом, валидация проекта показала, что вся функциональность была выполнена как было заявлено в задании.

При выполнении тестов 2-11 ошибок выявлено не было, выполнено успешно.

При тестировании программы не было выявлено ошибочных действий.

Протестированные функции работают безошибочно и никакие ошибочные действия пользователя не приведут к прекращению работы приложения.

# 6 Применение

## 6.1 Назначение программы

Программное средство «Автоматизированное рабочее место налогового инспектора» предназначено для того, чтобы учёт находящихся на складе товаров было комфортным.

Главное окно представлено на рисунке 6.1.

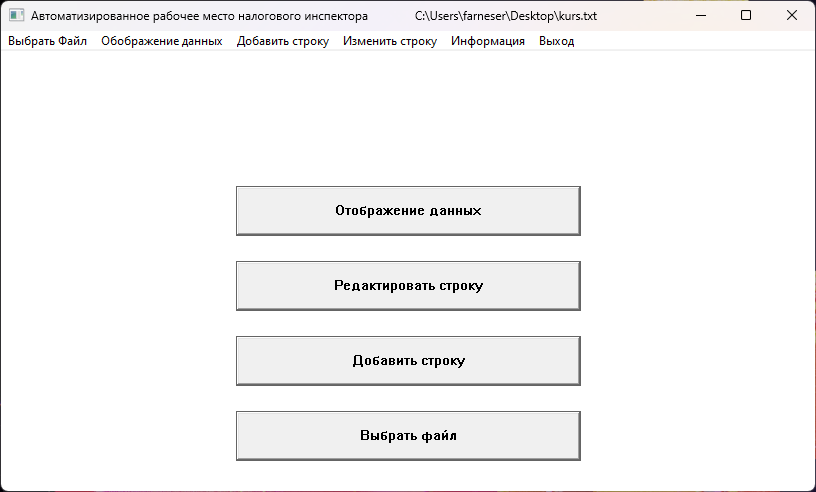


Рисунок 6.1 – Главное окно

## 6.2 Условия применения

Для проведения испытаний правильности работоспособности программы использовался компьютер со следующими характеристиками:

* Процессор AMD Ryzen 5 5600H 3.3ГГц;
* Объем оперативной памяти – 16 Гбайт;
* SSD накопитель формата NWMe 512 Гбайт.

Темой курсового проекта было проектирование «Автоматизированное рабочее место налогового инспектора».

Данное приложение создано, в первую очередь, для упрощения деятельности человека в учёте записей лиц, непрошедших проверку налогового инспектора.

Для начала работы с программой необходимо запустить файл Проект.exe. При запуске программы не требуется какая-либо авторизация пользователя для дальнейшего использования программы.

# Заключение

Заданием на курсовое проектирование являлась разработка программного приложения «Автоматизированное рабочее место налогового инспектора».

С этим программным обеспечением учёт материальных ценностей станет в разы проще и интереснее.

Основными задачами работы программного приложения являлись:

* ручное добавление записей в меню «Отображение данных»;
* ручное изменение записей в меню «Отображение данных»;
* ручное удаление записей в меню «Отображение данных»;
* просмотр готового списка записей путём чтения txt-файла через меню «Выбрать файл»;
* вывод информации в текстовой документ;
* просмотр готовой строчной таблицы с товарами через меню «Отображение данных»;

Результатом курсового проектирования явилась программа разработка программного приложения «Автоматизированное рабочее место налогового инспектора».

Преимуществами данной программы являются:

* быстрая и удобная навигация;
* удобство добавления товаров в общую базу;
* совместимость программы с различными операционными системами;
* минимальные требования к аппаратным ресурсам;
* удобный пользовательский интерфейс;
* целостность.

Программа реализована в соответствии со всеми заданными требованиями и нормами, отлажена и протестирована. Несоответствия с заданными требованиями в ходе работе не выявлено.

Программный продукт сочетается с потребностями обычного, неквалифицированного пользователя.

Задание на курсовое проектирование выполнено в полном объеме, все поставленные задачи реализованы и отвечают всем требованиям.

В дальнейшем программный продукт может дорабатываться, поскольку обладает средствами простого динамического обновления.

Пояснительная записка к курсовому проекту выполнена в соответствии с ЕСПД. Все ГОСТы были учтены.

# Список использованных источников

1. Т. Кормен: Алгоритмы: построение и анализ. –– Москва:ООО “Изд. Вильямс”, 2019.
2. С.М. Окулов: Программирование в алгоритмах, 2-е издание. – Москва:ООО “Бином-Пресс”, 2014.
3. А. Хортон: Visual C++. – Москва: ООО “Изд. Вильямс”, 2011.
4. Т.А. Павловская: C/C++. Программирование на языке высокого уровня – СПб: "Питер", 2008.
5. Т.Г. Багласова/К.О. Якимович: Методические указания по оформлению курсовых работ, дипломных проектов и отчётов. – Миснк: 2013

Нормативные правовые акты

1. ГОСТ 19.30F-2000 ЕСПД. Программа и методика испытаний. – Взамен ГОСТ 19.301-79 ; введ. в действие в Республике Беларусь 01.09.2001. – Минск : Белорус. гос. ин-т стандартизации и сертификации, 2000. – 9 с. – (Межгосударственный стандарт).
2. ГОСТ 19.401-2000 Единая система программной документации. Текст программы. Требования к содержанию, оформлению и контролю качества. – Взамен ГОСТ 19.401-78 ; введ. в действие в Республике Беларусь 01.09.2001. – Минск : Белорус. гос. ин-т стандартизации и сертификации, 2000. – 16 с. – (Межгосударственный стандарт).
3. ГОСТ 19.402-2000 Единая система программной документации. Описание программы. Требования к содержанию, оформлению и контролю качества. – Взамен ГОСТ 19.402-78 ; введ. в действие в Республике Беларусь 01.09.2001. – Минск : Белорус. гос. ин-т стандартизации и сертификации, 2000. – 20 с. – (Межгосударственный стандарт).
4. ГОСТ 2.105-95. Единая система конструкторской документации. Общие требования к текстовым документам. – Взамен ГОСТ 2.105-79 ГОСТ 2.906-71 ; введ. в действие в Республике Беларусь 30.06.1996**, в ред. 22.05.2013.** – Минск : Белорус. гос. ин-т стандартизации и сертификации, 2013. – 27 с. – (Межгосударственный стандарт).
5. ГОСТ 12.0.002-2003 Система стандартов безопасности труда. Термины и определения. – Взамен ГОСТ 12.0.002-80 ; введ. в действие в Республике Беларусь 12.06.2003**.** – Минск : Белорус. гос. ин-т стандартизации и сертификации, 2003. – 9 с. – (Межгосударственный стандарт).
6. ГОСТ 7.1-2003 Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. Общие требования и правила составления. – Взамен ГОСТ 7.1-84, ГОСТ 7.16-79, ГОСТ 7.18-79, ГОСТ 7.34-81, ГОСТ 7.40-82 ; введ. в действие в Республике Беларусь 01.11.2004**.** – Минск : Белорус. гос. ин-т стандартизации и сертификации, 2004. – 64 с. – (Госстандарт Республики Беларусь).

# Приложение А

Текст программы

(обязательное)

// kursach1.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "framework.h"

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)

{

TCHAR szFile[260] = { 0 };

for (int i = 0; i < 155; i++)

{

LIIIIIIIIIIIIIINE[i] = '-';

}

strcpy(path, "kurs.txt");

memset(&w, 0, sizeof(WNDCLASS));

w.style = CS\_HREDRAW | CS\_VREDRAW;

w.lpfnWndProc = WndProc;

w.hInstance = hInstance;

w.hbrBackground = CreateSolidBrush(0x00FFFFFF);

w.lpszClassName = "MyClass";

RegisterClass(&w);

hwnd = CreateWindow("MyClass", "Автоматизированное рабочее место налогового инспектора", WS\_OVERLAPPEDWINDOW, CW\_USEDEFAULT, 0, 630, 380, NULL, NULL, hInstance, NULL);

string titleP = path;

string titleA = "Автоматизированное рабочее место налогового инспектора\t";

string title = titleA + titleP;

SetWindowTextA(hwnd, title.c\_str());

ShowWindow(hwnd, nCmdShow);

UpdateWindow(hwnd);

while (GetMessage(&msg, NULL, 0, 0)) {

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return msg.wParam;

}

using namespace std;

LRESULT WINAPI WndProc(HWND hwnd, UINT Message, WPARAM wparam, LPARAM lparam) {

setlocale(LC\_ALL, "ru");

IFileOpenDialog\* pFileOpen;

root = CreateMenu();

hr = CoInitializeEx(NULL, COINIT\_APARTMENTTHREADED | COINIT\_DISABLE\_OLE1DDE);

hr = CoCreateInstance(CLSID\_FileOpenDialog, NULL, CLSCTX\_ALL, IID\_IFileOpenDialog, reinterpret\_cast<void\*\*>(&pFileOpen));

switch (Message) {

case WM\_CREATE:

hInst = ((LPCREATESTRUCT)lparam)->hInstance;

btnAddOK = CreateWindow("button", "Добавить", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

645, 360, 160, 30, hwnd, 0, hInst, NULL);

btnDeleteOK = CreateWindow("button", "Удалить", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

685, 10, 120, 30, hwnd, 0, hInst, NULL);

btnEditOK = CreateWindow("button", "Редактировать", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

645, 360, 160, 30, hwnd, 0, hInst, NULL);

btnBack = CreateWindow("button", "Назад", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

645, 400, 160, 30, hwnd, 0, hInst, NULL);

hStat = CreateWindow("static", "", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

10, 10, 625, 385, hwnd, 0, hInst, NULL);

hStatSEARCH = CreateWindow("static", "Отбор записей", WS\_CHILD | WS\_VISIBLE,

645, 50, 160, 20, hwnd, 0, hInst, NULL);

btnShow = CreateWindow("button", "Отображение данных", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

235, 135, 345, 50, hwnd, 0, hInst, NULL);

hStatMAIN = CreateWindow("static", mainTXT, WS\_CHILD | WS\_VISIBLE,

235, 30, 345, 80, hwnd, 0, hInst, NULL);

btnFile = CreateWindow("button", "Выбрать файл", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

235, 360, 345, 50, hwnd, 0, hInst, NULL);

btnDelete = CreateWindow("button", "Удалить строку", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

10, 50, 160, 30, hwnd, 0, hInst, NULL);

btnAdd = CreateWindow("button", "Добавить строку", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

235, 285, 345, 50, hwnd, 0, hInst, NULL);

btnEdit = CreateWindow("button", "Редактировать строку", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

235, 210, 345, 50, hwnd, 0, hInst, NULL);

btnSearch = CreateWindow("button", "Поиск", WS\_CHILD | WS\_VISIBLE | WS\_BORDER,

645, 110, 160, 30, hwnd, 0, hInst, NULL);

Edit1 = CreateWindow("edit", "", WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 10, 405, 55, 25, hwnd, 0, hInst, NULL);

Edit2 = CreateWindow("edit", "", WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 70, 405, 105, 25, hwnd, 0, hInst, NULL);

Edit3 = CreateWindow("edit", "", WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 180, 405, 105, 25, hwnd, 0, hInst, NULL);

Edit4 = CreateWindow("edit", "", WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 290, 405, 105, 25, hwnd, 0, hInst, NULL);

Edit5 = CreateWindow("edit", "", WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 400, 405, 160, 25, hwnd, 0, hInst, NULL);

Edit6 = CreateWindow("edit", "", WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 565, 405, 55, 25, hwnd, 0, hInst, NULL);

deleteEdit = CreateWindow("edit", "", WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 645, 10, 30, 30, hwnd, 0, hInst, NULL);

searchEdit = CreateWindow("edit", searchEditBase, WS\_CHILD | WS\_VISIBLE | WS\_BORDER, 645, 70, 160, 30, hwnd, 0, hInst, NULL);

AppendMenu(root, MF\_STRING, On\_Menu\_ClickedFILE, "Выбрать Файл");

AppendMenu(root, MF\_STRING, On\_Menu\_ClickedSHOW, "Обображение данных");

AppendMenu(root, MF\_STRING, On\_Menu\_ClickedADD, "Добавить строку");

AppendMenu(root, MF\_STRING, On\_Menu\_ClickedEDIT, "Изменить строку");

AppendMenu(root, MF\_STRING, On\_Menu\_ClickedABOUT, "Информация");

AppendMenu(root, MF\_STRING, On\_Menu\_ClickedEXIT, "Выход");

SetMenu(hwnd, root);

ShowWindow(Edit1, SW\_HIDE);

ShowWindow(Edit2, SW\_HIDE);

ShowWindow(Edit3, SW\_HIDE);

ShowWindow(Edit4, SW\_HIDE);

ShowWindow(Edit5, SW\_HIDE);

ShowWindow(Edit6, SW\_HIDE);

ShowWindow(btnFile, SW\_SHOWNORMAL);

ShowWindow(btnSearch, SW\_HIDE);

ShowWindow(hStatSEARCH, SW\_HIDE);

ShowWindow(searchEdit, SW\_HIDE);

ShowWindow(deleteEdit, SW\_HIDE);

ShowWindow(btnAddOK, SW\_HIDE);

ShowWindow(btnEditOK, SW\_HIDE);

ShowWindow(btnDeleteOK, SW\_HIDE);

ShowWindow(btnBack, SW\_HIDE);

ShowWindow(btnDelete, SW\_HIDE);

ShowWindow(btnShow, SW\_SHOWNORMAL);

ShowWindow(hStat, SW\_HIDE);

break;

case WM\_COMMAND:

switch (wparam)

{

case On\_Menu\_ClickedFILE:

lparam = (LPARAM)btnFile;

break;

case On\_Menu\_ClickedSHOW:

lparam = (LPARAM)btnShow;

break;

case On\_Menu\_ClickedEXIT:

PostQuitMessage(0);

break;

case On\_Menu\_ClickedADD:

lparam = (LPARAM)btnAdd;

break;

case On\_Menu\_ClickedEDIT:

lparam = (LPARAM)btnEdit;

break;

case On\_Menu\_ClickedABOUT:

MessageBox(NULL, "Данная программа разработана для курсового проекта по предмету \"Основы алгоритмизации и программирования\" учащегося Брестского филлиата Частного учреждения образования \"КОЛЛЕДЖ БИЗНЕСА И ПРАВА\", 2 курса, группы Т-091, Черновым Александром Евгеньевичем", "О программе", MB\_OK);

break;

default:

break;

}

if (lparam == (LPARAM)btnFile)

{

EnableWindow(hwnd, false);

hr = pFileOpen->Show(NULL);

if (SUCCEEDED(hr))

{

IShellItem\* pItem;

hr = pFileOpen->GetResult(&pItem);

if (SUCCEEDED(hr))

{

PWSTR pszFilePath;

hr = pItem->GetDisplayName(SIGDN\_FILESYSPATH, &pszFilePath);

wcstombs(path, pszFilePath, 256);

pItem->Release();

}

}

pFileOpen->Release();

EnableWindow(hwnd, true);

SetForegroundWindow(hwnd);

lparam = (LPARAM)btnShow;

string titleP = path;

string titleA = "Автоматизированное рабочее место налогового инспектора\t";

string title = titleA + titleP;

SetWindowTextA(hwnd, title.c\_str());

}

if (lparam == (LPARAM)btnSearch)

{

GetWindowText(searchEdit, Edit1TXT, 64);

searchSTR = Edit1TXT;

lparam = (LPARAM)btnShow;

}

if (lparam == (LPARAM)btnAddOK)

{

GetWindowText(Edit1, Edit1TXT, 20);

GetWindowText(Edit2, Edit2TXT, 20);

GetWindowText(Edit3, Edit3TXT, 20);

GetWindowText(Edit4, Edit4TXT, 20);

GetWindowText(Edit5, Edit5TXT, 20);

GetWindowText(Edit6, Edit6TXT, 20);

to\_add\_2 = Edit2TXT;

to\_add\_3 = Edit3TXT;

to\_add\_4 = Edit4TXT;

to\_add\_5 = Edit5TXT;

to\_add\_6 = Edit6TXT;

if (len(to\_add\_2) < 6) {

to\_add\_2 += "\t";

}

if (len(to\_add\_3) < 6) {

to\_add\_3 += "\t";

}

if (len(to\_add\_4) < 6) {

to\_add\_4 += "\t";

}

if (len(to\_add\_5) <= 14) {

to\_add\_5 += "\t";

if (len(to\_add\_5) <= 7)

{

to\_add\_5 += "\t";

}

}

ofstream out;

out.open(path, std::ios::app);

out << to\_add\_2 << "\t|" << to\_add\_3 << "\t|" << to\_add\_4 << "\t|" << to\_add\_5 << "\t|" << to\_add\_6 << "\t|" << endl;

out.close();

lparam = (LPARAM)btnShow;

}

if (lparam == (LPARAM)btnEditOK)

{

GetWindowText(Edit1, Edit1TXT, 64);

GetWindowText(Edit2, Edit2TXT, 64);

GetWindowText(Edit3, Edit3TXT, 64);

GetWindowText(Edit4, Edit4TXT, 64);

GetWindowText(Edit5, Edit5TXT, 64);

GetWindowText(Edit6, Edit6TXT, 64);

fstream in(path);

if (in.is\_open())

{

while (getline(in, line))

{

strings++;

if (maxLen < sizeof(line) / sizeof(line[0]))

{

maxLen = sizeof(line) / sizeof(line[0]);

}

}

}

in.close();

string\* mass = new string[strings];

in.open(path);

if (in.is\_open())

{

while (getline(in, line))

{

mass[fl] = line;

fl++;

}

}

in.close();

to\_add\_2 = Edit2TXT;

to\_add\_3 = Edit3TXT;

to\_add\_4 = Edit4TXT;

to\_add\_5 = Edit5TXT;

to\_add\_6 = Edit6TXT;

if (len(to\_add\_2) < 6) {

to\_add\_2 += "\t";

}

if (len(to\_add\_3) < 6) {

to\_add\_3 += "\t";

}

if (len(to\_add\_4) < 6) {

to\_add\_4 += "\t";

}

if (len(to\_add\_5) <= 14) {

to\_add\_5 += "\t";

if (len(to\_add\_5) <= 7)

{

to\_add\_5 += "\t";

}

}

mass[StrToInt(Edit1TXT) - 1] = to\_add\_2 + "\t|" + to\_add\_3 + "\t|" + to\_add\_4 + "\t|" + to\_add\_5 + "\t|" + to\_add\_6 + "\t|";

ofstream ofs;

ofs.open(path, std::ofstream::out | std::ofstream::trunc);

ofs.close();

ofstream file(path, ofstream::app | ofstream::out);

if (file.is\_open())

{

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

{

file << mass[i] << endl;

}

}

file.close();

lparam = (LPARAM)btnShow;

}

if (lparam == (LPARAM)btnDeleteOK)

{

GetWindowText(deleteEdit, lineToDel, 64);

lineNToDel = StrToInt(lineToDel) - 1;

fl = 0;

fl1 = 0;

countOfLinesInFile = 0;

fstream in(path);

if (in.is\_open())

{

while (getline(in, line))

{

countOfLinesInFile++;

}

}

in.close();

string\* mass = new string[countOfLinesInFile];

in.open(path);

if (in.is\_open())

{

while (getline(in, line))

{

if (fl1 != lineNToDel)

{

mass[fl] = line;

fl++;

}

fl1++;

}

}

in.close();

ofstream ofs;

ofs.open(path, std::ofstream::out | std::ofstream::trunc);

ofs.close();

ofstream file(path, ofstream::app | ofstream::out);

if (file.is\_open())

{

for (int i = 0; i < countOfLinesInFile - 1; i++)

{

file << mass[i] << endl;

}

}

file.close();

lparam = (LPARAM)btnShow;

}

if (lparam == (LPARAM)btnAdd)

{

ShowWindow(btnShow, SW\_HIDE);

ShowWindow(btnFile, SW\_HIDE);

ShowWindow(btnAdd, SW\_HIDE);

ShowWindow(btnEdit, SW\_HIDE);

ShowWindow(btnDelete, SW\_HIDE);

ShowWindow(btnEditOK, SW\_HIDE);

SetWindowText(Edit1, "");

SetWindowText(Edit2, "Фамилия");

SetWindowText(Edit3, "Имя");

SetWindowText(Edit4, "Город");

SetWindowText(Edit5, "Улица");

SetWindowText(Edit6, "№Дом");

countOfLinesInFile = 0;

ifstream in(path);

if (in.is\_open())

{

while (getline(in, line))

{

++countOfLinesInFile;

}

}

in.close();

lparam = (LPARAM)btnShow;

ShowWindow(btnAddOK, SW\_SHOWNORMAL);

ShowWindow(btnBack, SW\_SHOWNORMAL);

ShowWindow(Edit1, SW\_HIDE);

ShowWindow(Edit2, SW\_SHOWNORMAL);

ShowWindow(Edit3, SW\_SHOWNORMAL);

ShowWindow(Edit4, SW\_SHOWNORMAL);

ShowWindow(Edit5, SW\_SHOWNORMAL);

ShowWindow(Edit6, SW\_SHOWNORMAL);

}

if (lparam == (LPARAM)btnEdit)

{

ShowWindow(btnShow, SW\_HIDE);

ShowWindow(btnFile, SW\_HIDE);

ShowWindow(btnEdit, SW\_HIDE);

ShowWindow(btnAdd, SW\_HIDE);

ShowWindow(btnAddOK, SW\_HIDE);

ShowWindow(btnDelete, SW\_HIDE);

SetWindowText(Edit1, "");

SetWindowText(Edit2, "");

SetWindowText(Edit3, "");

SetWindowText(Edit4, "");

SetWindowText(Edit5, "");

SetWindowText(Edit6, "");

countOfLinesInFile = 0;

ifstream in(path);

if (in.is\_open())

{

while (getline(in, line))

{

++countOfLinesInFile;

}

}

in.close();

ShowWindow(btnEditOK, SW\_SHOWNORMAL);

ShowWindow(btnBack, SW\_SHOWNORMAL);

ShowWindow(Edit1, SW\_SHOWNORMAL);

ShowWindow(Edit2, SW\_SHOWNORMAL);

ShowWindow(Edit3, SW\_SHOWNORMAL);

ShowWindow(Edit4, SW\_SHOWNORMAL);

ShowWindow(Edit5, SW\_SHOWNORMAL);

ShowWindow(Edit6, SW\_SHOWNORMAL);

lparam = (LPARAM)btnShow;

}

if (lparam == (LPARAM)btnShow)

{

ShowWindow(btnShow, SW\_HIDE);

ShowWindow(btnFile, SW\_HIDE);

ShowWindow(btnAdd, SW\_HIDE);

ShowWindow(btnEdit, SW\_HIDE);

ShowWindow(btnDelete, SW\_HIDE);

ShowWindow(hStatMAIN, SW\_HIDE);

ShowWindow(hStat, SW\_SHOWNORMAL);

ShowWindow(searchEdit, SW\_SHOWNORMAL);

ShowWindow(btnSearch, SW\_SHOWNORMAL);

ShowWindow(hStatSEARCH, SW\_SHOWNORMAL);

ShowWindow(btnBack, SW\_SHOWNORMAL);

ShowWindow(btnDeleteOK, SW\_SHOWNORMAL);

ShowWindow(deleteEdit, SW\_SHOWNORMAL);

strcpy(strExit, "");

int fl = 0;

ifstream in(path);

if (in.is\_open())

{

strcat(strExit, "№\t|Фамилия\t|Имя\t\t|Назв.города\t|Назв.улицы\t\t|№Дом\t|");

strcat(strExit, "\n");

strcat(strExit, LIIIIIIIIIIIIIINE);

strcat(strExit, "\n");

while (getline(in, line))

{

++fl;

if (isStrInLine(line, searchSTR)) {

strcat(strExit, IntToStr(fl));

strcat(strExit, "\t|");

strcat(strExit, line.c\_str());

strcat(strExit, "\n");

}

}

}

in.close();

SetWindowTextA(hStat, strExit);

}

if (lparam == (LPARAM)btnBack)

{

ShowWindow(btnShow, SW\_SHOWNORMAL);

ShowWindow(btnFile, SW\_SHOWNORMAL);

ShowWindow(btnAdd, SW\_SHOWNORMAL);

ShowWindow(btnEdit, SW\_SHOWNORMAL);

ShowWindow(hStat, SW\_HIDE);

ShowWindow(btnBack, SW\_HIDE);

ShowWindow(btnAddOK, SW\_HIDE);

ShowWindow(btnEditOK, SW\_HIDE);

ShowWindow(btnDeleteOK, SW\_HIDE);

ShowWindow(deleteEdit, SW\_HIDE);

ShowWindow(btnSearch, SW\_HIDE);

ShowWindow(hStatSEARCH, SW\_HIDE);

ShowWindow(searchEdit, SW\_HIDE);

ShowWindow(Edit1, SW\_HIDE);

ShowWindow(Edit2, SW\_HIDE);

ShowWindow(Edit3, SW\_HIDE);

ShowWindow(Edit4, SW\_HIDE);

ShowWindow(Edit5, SW\_HIDE);

ShowWindow(Edit6, SW\_HIDE);

}

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

case WM\_GETMINMAXINFO:

{

LPMINMAXINFO lpMMI = (LPMINMAXINFO)lparam;

lpMMI->ptMinTrackSize.x = windowSizeX;

lpMMI->ptMinTrackSize.y = windowSizeY;

lpMMI->ptMaxTrackSize.x = windowSizeX;

lpMMI->ptMaxTrackSize.y = windowSizeY;

}

default:

return DefWindowProc(hwnd, Message, wparam, lparam);

}

return 0;

} // код нуждается в оптимизации, но и так работает

// framework.h

#pragma once

#include "targetver.h"

#define WIN32\_LEAN\_AND\_MEAN

#include <windows.h>

#include <stdlib.h>

#include <malloc.h>

#include <memory.h>

#include <tchar.h>

#include "kursach1.h"

#include <string>

#include <fstream>

#include <windows.h>

#include <string>

#include <commdlg.h>

#include <shobjidl.h>

#include "func.h"

#include "perem.h"

#include <Lmcons.h>

// func.h

#pragma once

#include "framework.h"

#include <string>

int StrToInt(char\* s)

{

int temp = 0;

int i = 0;

int sign = 0;

if (s[i] == '-')

{

sign = 1;

i++;

}

while (s[i] >= 0x30 && s[i] <= 0x39)

{

temp = temp + (s[i] & 0x0F);

temp = temp \* 10;

i++;

}

temp = temp / 10;

if (sign == 1)

temp = -temp;

return(temp);

}

char\* IntToStr(int n)

{

char s[40], t, \* temp;

int i, k;

int sign = 0;

i = 0;

k = n;

if (k < 0)

{

sign = 1;

k = -k;

}

do {

t = k % 10;

k = k / 10;

s[i] = t | 0x30;

i++;

} while (k > 0);

if (sign == 1)

{

s[i] = '-';

i++;

}

temp = new char[i];

k = 0;

i--;

while (i >= 0) {

temp[k] = s[i];

i--; k++;

}

temp[k] = '\0';

return(temp);

}

int len(std::string strEAXCSAE) {

int length = 0;

for (int i = 0; strEAXCSAE[i] != '\0'; i++)

{

length++;

}

return length;

}

using namespace std;

string\* split(string str, char\* seperator)

{

string\* array = new string[6];

int currIndex = 0, i = 0;

int startIndex = 0, endIndex = 0;

while (i <= len(str) - 4)

{

if (str[i] == str[i] == seperator[0] || seperator[1] || str[i] == seperator[2] || str[i] == seperator[3] || i == len(str))

{

endIndex = i;

string subStr = "";

subStr.append(str, startIndex, endIndex - startIndex);

array[currIndex] = subStr;

currIndex += 1;

startIndex = endIndex + 1;

}

i++;

}

return array;

}

int len(TCHAR\* str) {

return sizeof(str) / sizeof(str[0]);

}

BOOL isStrInLine(std::string line, string search) {

if (line.find(search) <= line.length())

{

return true;

}

return false;

}

// kursach1.h

#pragma once

#include "resource.h"

#include "Resource.h"

// perem.h

#pragma once

#include "framework.h"

HINSTANCE hInst;

WCHAR szTitle[MAX\_LOADSTRING];

WCHAR szWindowClass[MAX\_LOADSTRING];

HWND hwnd;

MSG msg;

WNDCLASS w;

HRESULT hr;

HDC hdc;

PAINTSTRUCT ps;

HMENU root;

ATOM MyRegisterClass(HINSTANCE hInstance);

BOOL InitInstance(HINSTANCE, int);

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

static HWND hEdt1, hEdt2, hStat, hStatMAIN, hStatSEARCH, hList, btnBack, btnAddOK,

btnDeleteOK, btnRefresh, btnAdd, btnShow, btnFile, btnDelete, btnEdit, btnSearch,

btnAbout, btnEditOK, Edit1, Edit2, Edit3, Edit4, Edit5, Edit6, deleteEdit, searchEdit;

string to\_add\_2, to\_add\_3, to\_add\_4, to\_add\_5, to\_add\_6, line, searchSTR = "|";

char\* strIn = new char[8095];

char\* strExit = new char[8095];

int fl = 0, fl1 = 0, strings = 0, maxLen = 0, lineNToDel, countOfLinesInFile;

char seperator[] = "&^)\*", LIIIIIIIIIIIIIINE[155], path[256] = "";

TCHAR Edit1TXT[64], Edit2TXT[64], Edit3TXT[64], Edit4TXT[64], Edit5TXT[64], Edit6TXT[64], lineToDel[64];

char mainTXT[] = "Вас приветствует мастер работы налогового инспектора\ngithub.com/farneser\n(курсовая работа появистя там только после сдачи)";

TCHAR\* searchEditBase = (TCHAR\*)"";

// Resource.h

//{{NO\_DEPENDENCIES}}

#define IDS\_APP\_TITLE 103

#define IDR\_MAINFRAME 128

#define IDD\_KURSACH1\_DIALOG 102

#define IDD\_ABOUTBOX 103

#define IDM\_ABOUT 104

#define IDM\_EXIT 105

#define IDI\_KURSACH1 107

#define IDI\_SMALL 108

#define IDC\_KURSACH1 109

#define IDC\_MYICON 2

#ifndef IDC\_STATIC

#define IDC\_STATIC -1

#define MAX\_LOADSTRING 100

#define windowSizeX 830

#define windowSizeY 499

#define MAX\_COUNT\_OF\_SPLITS 6

#define On\_Menu\_ClickedFILE 3370

#define On\_Menu\_ClickedSHOW 3371

#define On\_Menu\_ClickedEXIT 3372

#define On\_Menu\_ClickedADD 3373

#define On\_Menu\_ClickedEDIT 3374

#define On\_Menu\_ClickedTEST 3375

#define On\_Menu\_ClickedABOUT 3376

#endif

#ifdef APSTUDIO\_INVOKED

#ifndef APSTUDIO\_READONLY\_SYMBOLS

#define \_APS\_NO\_MFC 130

#define \_APS\_NEXT\_RESOURCE\_VALUE 129

#define \_APS\_NEXT\_COMMAND\_VALUE 32771

#define \_APS\_NEXT\_CONTROL\_VALUE 1000

#define \_APS\_NEXT\_SYMED\_VALUE 110

#endif

#endif

// targetver.h

#pragma once

#include <SDKDDKVer.h>