Частное учреждение образования

«Колледж бизнеса и права»

СОЗДАНИЕ ПРОГРАММНОГО СРЕДСТВА ПО УЧЕБНОМУ ПРЕДМЕТУ ИНСТРУМЕНТАЛЬНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ ИЗУЧЕНИЯ ТЕОРЕТИЧЕСКОГО МАТЕРИАЛА И РЕШЕНИЯ БАЗОВЫХ ЗАДАЧ С ВОЗМОЖНОСТЬЮ ПРОВЕРКИ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине

«Основы алгоритмизации и программирования»

КП Т.295018.401

Руководитель проекта (Дж. А. Миронова)

Учащийся (М. А. Симанович)

2024

Изм.

Лист

№ докум.

Подпись

Датаа

Лист

3

КП Т.295018.401 ПЗ

Разраб.

Симанович М.А.

Провер.

Миронова Дж.А.

Реценз.

Н. Контр.

Утверд.

Создание программного средства по учебному предмету «Инструментальное программное обеспечение» для изучения теоретического материала и решения базовых задач с возможностью проверки

Лит.

Листов

КБиП

50

У

**СОДЕРЖАНИЕ**

[Введение 4](#_Toc170314480)

[1 Описание задачи 5](#_Toc170314481)

[1.1 Анализ предметной области 5](#_Toc170314482)

[1.2 Постановка задачи 6](#_Toc170314483)

[2 Проектирование системы 7](#_Toc170314484)

[2.1 Требования к приложению 7](#_Toc170314485)

[2.2 Проектирование модели 7](#_Toc170314486)

[2.3 Концептуальный прототип 8](#_Toc170314487)

[3 Описание реализации программного средства 11](#_Toc170314488)

[3.1 Инструменты разработки и применяемые технологии 11](#_Toc170314489)

[3.2 Функции: логическая и физическая организация 11](#_Toc170314490)

[3.3 Входные и выходные данные 15](#_Toc170314491)

[3.4 Функциональное тестирование 15](#_Toc170314492)

[3.5 Описание справочной системы 17](#_Toc170314493)

[4 Применение 18](#_Toc170314494)

[4.1 Назначение программного средства 18](#_Toc170314495)

[4.2 Условия применения 18](#_Toc170314496)

[Заключение 19](#_Toc170314497)

[Список использованных источников 20](#_Toc170314498)

[Приложение А 21](#_Toc170314499)

[Приложение Б 48](#_Toc170314500)

Введение

В современном мире трудно представить жизнь без IT-технологий. В этой цифровой эпохе использование программного обеспечения становится неотъемлемой частью нашей реальности. Все ищут способы попасть в IT, выучить язык программирования и вложить свой вклад в развитие технологий. Это инструмент, который облегчает обучение, улучшает процессы обучения и управления информацией, а также способствует развитию профессиональных навыков.

Важность создания программного средства трудно переоценить. Этот инструмент не только обеспечит удобное управление учебным материалом и выполнение практических заданий, но и предоставит пользователю возможность эффективной работы с информацией и обмена опытом с коллегами.

Целью разработки курсового проекта на тему «Создание программного средства по учебному предмету Инструментальное программное обеспечение» является реализация инструмента, который позволит студентам учиться эффективно, проверять свои знания и обмениваться опытом с коллегами, благодаря удобному интерфейсу, возможности выполнять практические задания, формированию отчетов, изучению и пометок в документации.

Актуальность программного средства заключается в том, что оно отвечает современным требованиям к образованию и управлению информацией, предоставляя пользователю инновационные инструменты для обучения и развития профессиональных навыков.

В первом разделе «Описание задачи» основное внимание уделяется выявлению ключевых задач, необходимых для успешной реализации проекта. Это включает в себя выявление основных функциональных требований к программному средству, а также план разработки.

Во втором разделе «Проектирование системы» представлено описание структуры программного средства, включая создание лаконичного интерфейса с использованием меню, кнопочных форм и панелей инструментов, а также организацию иерархии классов и работы с информацией.

В третьем разделе «Реализация программного средства» затронуты технические аспекты создания программного средства, включая выбор используемых технологий, алгоритмы и архитектуру приложения, а также реализацию всех функций и возможностей, описанных в задании.

Четвертый раздел «Применение» посвящен обзору возможностей и сфер применения разработанного программного средства, а также его влиянию на повседневную деятельность пользователей, обучение и управление информацией.

В Заключении проведен анализ результатов разработки, выделены основные достижения и определены перспективы дальнейшего развития проекта.

1 Описание задачи

1.1 Анализ предметной области

Целью разработки курсового проектирования на тему «Создание программного средства по учебному предмету Инструментальное программное обеспечение» является реализация программного средства, которое предоставляет пользователям возможность входа в личный аккаунт или создания нового, просмотра доступных учебных материалов, регистрации в приложении, добавления материалов в избранное и получения краткого описания учебных курсов.

Предметная область языка программирования Python охватывает широкий спектр применений, что делает его одним из самых популярных языков в мире. Python используется во множестве сфер, например В области машинного обучения и искусственного интеллекта Python занимает центральное место, поддерживая такие инструменты как TensorFlow, Keras и PyTorch для разработки и обучения моделей машинного обучения. Python также используется для автоматизации задач и написания скриптов, облегчая системное администрирование и рутинные задачи. В разработке игр Python используется с библиотекой Pygame для создания 2D игр и прототипирования игровых концепций и многое другое.

Компьютерная обработка значительно упрощает процесс доступа к учебным материалам по сравнению с реальным визитом. Она повышает спрос пользователей и их заинтересованность, обеспечивает быстрый доступ к доступным курсам и их стоимости.

Потенциальными пользователями программного средства являются студенты, поскольку учебные курсы востребованы среди молодежи. Программное средство разработано с целью объединения всех необходимых средств для изучения языка программирования.

1.2 Постановка задачи

Задачей проекта является разработка программного средства для просмотра и прохождения учебных курсов. Программа обеспечивает просмотр и ознакомление с курсами, их прохождения, просмотр видеороликов и прохождения практических заданий.

Программное средство будет включать в себя функционал для поиска по названию курса, что позволяет пользователям ориентироваться в большом объеме данных. Также будет возможность сохранения и просмотра пройденных курсов.

На рисунке 1.1 представлен аналог разрабатываемого программного средства «Stepik».

Рисунок 1.1 «Stepik»

Достоинства аналога:

1. Интерфейс: Простой и интуитивно понятный интерфейс облегчает навигацию и использование приложения.

2. Доступность: Возможность регистрации в приложении в любое время и из любого места, используя персональный компьютер

3. История прохождения курсов и отчеты: Хранение истории пройденных курсов, возможность создания отчетов

Недостатки аналога:

1. Зависимость от интернета: Необходимость постоянного подключения к интернету может ограничивать использование приложения в условиях слабого или отсутствующего интернета.

2. Технические проблемы: Возможные ошибки и сбои в работе приложения могут вызвать неудобства и снизить доверие пользователей.

3. Актуальность информации: Необходимость постоянного обновления информации о доступных курсах и их расписании может потребовать значительных усилий от организаторов.

Разрабатываемое программное средство будет включать в себя различные особенности, позволяющие более комфортно взаимодействовать с приложением.

2 Проектирование системы

2.1 Требования к приложению

Безопасность в приложении обеспечена на высоком уровне односторонним шифрованием в хэш и записью в бинарные файлы (.bin). В такие файлы записываются логин, пароль, личные данные пользователя из полей. В приложении реализована авторизация и регистрация, она локальная, потому что приложение поддерживается никакими онлайн серверами и является полностью локальным, что еще больше минимизирует риски взломов.

Интерфейс лаконичен и интуитивно понятен, он не требует от пользователя специальных технических знаний; все элементы управления расположены логично и последовательно. Реализована система вкладок в основной форме, чтобы не нагружать пользователя новыми дополнительными окнами.

Для удобства работы пользователей размеры кнопок и полей ввода, используется современный, читаемый шрифт, Century Gothic размером от 8 до 25 пунктов. Цветовая схема интерфейса темно-синего оттенка, что уменьшает нагрузку на глаза при длительной работе. Применяются сторонние элементы управления из специальных фреймворков, такие как закругленные, анимированные кнопки, выпадающие закругленные списки и закругленные поля ввода, соответствующие принципам удобства использования и красоты.

Минимальные требования к аппаратным и операционным ресурсам:

* OC: Windows 10
* Процессор: Pentium® 4 1.5 GHz / Athlon® XP
* Оперативная память: 2048 MB ОЗУ
* Видеокарта: Встроенная графика или любая дискретная видеокарта.

2.2 Проектирование модели

Для отображения функциональной структуры объектов программного средства необходимо воспользоваться унифицированным языком моделирования UML, а именно диаграмму классов и диаграмму работы алгоритмов.

Диаграмма работы алгоритмов — это структурная диаграмма в языке UML является графическим представлением последовательности шагов выполнения алгоритма. Она помогает визуализировать логику алгоритма и его структуру, делая процесс понятным и легко читаемым. Данная диаграмма полезна при проектировании, отладке и понимании алгоритмов. Они помогают программистам и разработчикам лучше организовать свои мысли, выделить ключевые шаги и легко обнаружить ошибки или улучшить производительность алгоритма. Диаграмма работы алгоритмов представлена в графической части на листе 1.

Диаграмма классов — это структурная диаграмма в языке UML, которая отображает структуру системы, описывая классы, их атрибуты, методы и отношения между ними. Диаграмма классов помогает визуализировать архитектуру программного обеспечения и отношения между объектами. Данная диаграмма помогает разработчикам лучше понять структуру системы, улучшить проектирование программного обеспечения и облегчить коммуникацию между участниками проекта. Они являются важным инструментом для проектирования и документирования программных систем. Диаграмма классов представлена в графической части на листе 2.

2.3 Концептуальный прототип

Концептуальный прототип – это общее видение продукта, его функций, логики взаимодействия с другими компонентами. Цель концептуального прототипа — получить обратную связь от пользователей и заинтересованных сторон на ранних этапах разработки, что позволяет уточнить требования и улучшить дизайн до начала полноценной разработки.

Окно авторизации включает в себя список кнопок, которые выполняют различные действия. Графическое представление прототипа окна авторизации представлено на рисунке 2.1.

Рисунок 2.1 – прототип окна авторизации

Кнопка регистрация оставляет пользователя на этой форме, добавляет кнопку зарегистрироваться и textBox для подтверждения пароля. Прототип представлен на рисунке 2.2.

Рисунок 2.2 – окно регистрации

Кнопка вход перебрасывает пользователя на рабочее окно приложения. Прототип рабочего окна приложения представлен на рисунке 2.3.

Рисунок 2.3 – прототип рабочего окна приложения

После нажатия кнопки регистрации идет перенаправление на окно заполнения профиля. Где он должен подтвердить свой email и указать личные данные. Прототип предоставлен на рисунке 2.4

Рисунок 2.4 – прототип окна заполнения профиля

3 Описание реализации программного средства

3.1 Инструменты разработки и применяемые технологии

Разработка приложения на С++ с использованием фреймворков Guna2UI, SiticoneDesktop и BunifuFramework в среде Visual Studio представляет собой процесс, направленный на создание высокопроизводительного клиентского приложения с современным и эстетичным пользовательским интерфейсом.

Язык программирования C++ выбран из-за его высокой производительности и возможности эффективного управления ресурсами, что особенно важно для приложений, взаимодействующих с данными из интернета, такими как данные о криптовалютах. Visual Studio, в качестве интегрированной среды разработки, обеспечивает удобную платформу для создания, отладки и развертывания приложений на C++. Она интегрирует в себя все необходимые инструменты и ресурсы для эффективной работы над проектами.

Фреймворки Guna2UI, SiticoneDesktop и BunifuFramework предоставляют разнообразные графические компоненты и элементы управления, которые позволяют создавать современные пользовательские интерфейсы. Они предоставляют не только стандартные элементы WinForms, но и расширенные возможности для стилизации и анимации интерфейса, что значительно повышает его привлекательность и удобство использования для конечного пользователя.

Такой подход позволяет разработать полноценное обучающее приложение, способное эффективно обрабатывать код, предоставляя пользователю удобный и современный интерфейс для взаимодействия с документацией, задачами и другими данными.

3.2 Функции: логическая и физическая организация

Реализация системы регистрации. Эта функция выполняется на стадии регистрации логина и пароля, она проверяет поля на корректность, в случае если какое-либо поле некорректно, она завершает свою работу. В ином случае функция вызывает другую, которая записывает в файл данные.

Void auth::buttonRegister\_Click(Object^ sender, EventArgs^ e) {

// Проверка на пустые поля

if (textBoxLogin1->Text == "") {

textBoxLogin1->BorderColor = Color::Red;

MessageWarning->Show("Введите логин");

return;

}

if (textBoxPassword1->Text == "" || textBoxPassword\_1->Text == "") {

textBoxPassword1->BorderColor = Color::Red;

textBoxPassword\_1->BorderColor = Color::Red;

MessageWarning->Show("Введите пароль");

return;

}

// Проверка на совпадение паролей

if (textBoxPassword1->Text != textBoxPassword\_1->Text) {

textBoxPassword1->BorderColor = Color::Red;

textBoxPassword\_1->BorderColor = Color::Red;

MessageError->Show("Пароли не совпадают", "Ошибка");

return;

}

// Проверка на соглашение

if (checkBoxTerms->Checked == false) {

MessageError->Caption = "Предупреждение";

MessageError->Text = "Для регистрации вам нужно прочесть условия использования";

MessageError->Show();

return;

}

// Проверка на существование уже такого пользователя

if (Directory::Exists(textBoxLogin1->Text)) {

MessageError->Caption = "Предупреждение";

MessageError->Text = "Пользователь под таким логином уже существует";

MessageError->Show();

return;

}

// Регистрация в случае если все условия соблюдены

panelMain->Visible = false;

Register();

}

Реализация функции Register(). Эта функция сохраняет логин, пароль пользователя.

Void auth::Register() {

// Получаем логин и пароль пользователя

String^ cliLogin = textBoxLogin1->Text;

String^ cliPassword = textBoxPassword1->Text;

// При получении данных меняем данные формы

USER = cliLogin;

labelWelcome->Text += cliLogin;

SetCenter(pageWelcome, labelWelcome, 1);

std::string Login = msclr::interop::marshal\_as<std::string>(cliLogin);

std::string Password = msclr::interop::marshal\_as<std::string>(cliPassword);

// Создаем папку с полученным логином

String^ folderPath = cliLogin;

Directory::CreateDirectory(folderPath);

// Путь к файлу Password - data.bin

String^ filePath = Path::Combine(folderPath, "data.bin");

// Хэшируем пароль

std::hash<std::string> hasher;

size\_t hashPassword = hasher(Password);

// Открываем файл для записи в бинарном режиме

FileStream^ fileStream = gcnew FileStream(filePath, FileMode::Create, FileAccess::Write);

BinaryWriter^ binaryWriter = gcnew BinaryWriter(fileStream);

binaryWriter->Write(hashPassword);

binaryWriter->Close();

fileStream->Close();

// Анимация, переход на следующую страницу.

animation\_mode = 1;

sec = 4;

timerTransition->Start();

}

Реализация функции входа в приложение:

Обработчик события кнопки «Войти» проверяет корректность введенных данных в поля, активировал ли пользователь функцию запоминания данных при последующих входах, и вызывает функцию входа типа Boolean, которая проверяет данные сохранённые в файлы и введенные пользователем. В случае сходства данных функция возвращает true иначе false.

Если функция возвращает true, обработчик активирует анимацию перехода на основную форму. Приветствие показывается если пользователь включил это в настройках.

Void auth::buttonComeIn\_Click(Object^ sender, EventArgs^ e) {

if (textBoxLogin->Text == "") {

textBoxLogin->BorderColor = Color::Red;

MessageWarning->Show("Введите логин");

return;

}

if (textBoxPassword->Text == "") {

textBoxPassword->BorderColor = Color::Red;

MessageWarning->Show("Введите пароль");

return;

}

if (ComeIn()) {

if (SwitchRemember->Checked == true) {

CreateLogFile();

}

if (greeting) {

animation\_mode = 4;

sec = 4;

}

else {

animation\_mode = 5;

sec = 1;

}

labelWelcome->Text += USER;

SetCenter(pageWelcome, labelWelcome, 1);

timerTransition->Start();

}

}

Boolean auth::ComeIn() {

// Получаем логин и пароль пользователя

String^ cliLogin = textBoxLogin->Text;

String^ cliPassword = textBoxPassword->Text;

// Преобразуем их в std::string

std::string Login = msclr::interop::marshal\_as<std::string>(cliLogin);

std::string Password = msclr::interop::marshal\_as<std::string>(cliPassword);

// Создаем путь к файлу Password - data.bin

String^ folderPath = cliLogin;

String^ filePath = Path::Combine(folderPath, "data.bin");

// Проверяем существует ли пользователь в бд

if (!Directory::Exists(folderPath)) {

MessageError->Caption = "Ошибка";

MessageError->Text = "Пользователя с указанным логином не существует";

MessageError->Show();

return false;

}

// Хэшируем введенный пароль

std::hash<std::string> hasher;

size\_t hashPasswordInput = hasher(Password);

// Открываем файл для чтения в бинарном режиме

FileStream^ fileStream = gcnew FileStream(filePath, FileMode::Open, FileAccess::Read);

BinaryReader^ binaryReader = gcnew BinaryReader(fileStream);

// Читаем данные из файла

size\_t hashPasswordFile = binaryReader->ReadUInt64();

// Закрываем бинарный читатель и файловый поток

binaryReader->Close();

fileStream->Close();

// Сравниваем хэши паролей

if (hashPasswordInput == hashPasswordFile) {

USER = cliLogin;

return true; // Пароль совпадает

}

else {

MessageError->Caption = "Предупреждение";

MessageError->Text = "Введен неверный пароль";

MessageError->Show();

return false; // Пароль не совпадает

}

}

Текст остальных программных модулей представлен в приложении А.

3.3 Входные и выходные данные

Входные данные:

1. Логин и пароль: Строка, содержащая логин из букв и (или) цифр.
2. Имя и Фамилия: Строка, содержащая имя пользователя из букв и (или) цифр.
3. Дата рождения: Выбранные элементы из массива в элементе dropBox.
4. Электронная почта: Строка, являющаяся адресом, которая имеет ключевые символы.
5. Исходный код на Python: Строка, содержащая код задачи на языке Python.
6. Аватар пользователя: Изображение, имеющее jpg или png формат.

Выходные данные:

1. Логин: Папка, имеющая наименование равное логину пользователя
2. Файл с паролем: Хешированная строка, записанная в бинарном файле.
3. Файл с датой рождения: Дата, содержащая информацию о дате рождения пользователя, записанная в бинарном виде.
4. Файл с именем: Строка, содержащая имя пользователя.
5. Файл с полом: Строка, содержащая пол пользователя.
6. ­Файл с фамилией: Строка, содержащая фамилию пользователя, если он её указывал.
7. Файл с прогрессом: Число, преобразованное в строку и записанная в бинарный файл с помощью специального ключа шифрования.
8. Файл с уровнем: Строка, содержащая информацию о уровне пользователя.
9. Файл с состоянием задач: Словарь, содержащий информацию о состояния решения каждой задачи.
10. Тесты для задачи: Скрипт, содержащий Unit-тесты написанные на Python, который проверяет исходный код пользователя на Python.
11. Файл отчета скрипта: Строка, сообщающая программе результаты тестов.
12. Файл решения: Отчёт, содержащий код решения задачи пользователя.

3.4 Функциональное тестирование

Функциональное тестирование – это вид тестирования, который проверяет соответствие функциональности продукта тому, как он был задуман. Оценка производится в соответствии с ожидаемыми и полученными результатами (на основании функциональной спецификации), при условии, что функции отрабатывали на различных значениях.

Тест-кейсы для основных функций представлены в таблице 3.1.

Таблица 3.1 – Функциональное тестирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №№ | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
|  | Авторизация пользователей | 1. Ввести существующие и верные логин и пароль. 2. Нажать кнопку «Войти» | Открытие главного окна на странице «Главная» | Результат представлен на рисунке В.1 и В.2 |
|  | Функция «Запомнить меня» | 1. Ввести существующие и верные логин и пароль. 2. Включение функции нажатием на кнопку «Запомнить меня». 3. Нажать кнопку «Войти» |  |  |
|  | Регистрация пользователей | 1. Ввести корректные логин, пароль и пароль повторно. 2. Нажать на кнопку «Регистрация» |  |  |
|  | Просмотр личной информации пользователя | 1. Выбрать страницу «Профиль». 2. Ознакомиться с информацией |  |  |
|  | Редактирование личной информации пользователя | 1. Выбрать страницу «Профиль» 2. Нажать на кнопку «Редактировать» 3. Редактировать разрешенные поля |  |  |
|  | Редактирование Email | 1. Выбрать страницу «Профиль» 2. Нажать на кнопку «Редактировать» 3. Изменить Email в поле 4. Нажать кнопку «Отправить письма» 5. Ввести первый полученный с ранее привязанной почты код в поле «Код с старой почты» 6. Ввести второй полученный код в поле «Код с новой почты» 7. Нажать кнопку «Подтвердить» |  |  |
|  | Смена пароля | 1. Ввести действительный пароль в поле «Старый пароль» 2. Ввести новый пароль в поле «Новый пароль» 3. Ввести новый пароль еще раз в поле «Повтор пароля» 4. Нажать кнопку «Изменить» |  |  |
|  | Удаление аккаунта | 1. Ввод пароля в поле «Старый пароль» 2. Нажать на гиперссылку «Хотите удалить аккаунт?» 3. Подтвердить действие |  |  |
|  | Переход на другую страницу без сохранения профиля | 1. Изменить данные профиля 2. Перейти на другую страницу 3. Подтвердить действие |  |  |
|  | Изменение настроек | 1. Выбрать страницу «Настройки» 2. Изменить значения 3. Нажать кнопку «Сохранить» |  |  |
|  | Посетить страницу «Задания» впервые | 1. Выбрать страницу «Задания» 2. Просмотреть анимацию 3. Подтвердить ознакомление с политикой задач |  |  |
|  | Реализация теоретического материала по курсам, с возможностью создания пометок в предоставленном материале |  |  | Результат представлен на рисунке В.6, В.7 и В.8. |
|  | Реализация возможности прохождения практических заданий в виде решения задач при помощи интерпретатора Python и предоставление результата при о пройденных заданиях. |  |  | Результат представлен на рисунке В.9 и В.10 |
|  | Осуществление формирования отчетов прохождения практических заданий за выбранный промежуток времени. |  |  |  |
|  | Экспортирование отчетов |  |  |  |
|  | Организация лаконичного интерфейса программного средства с помощью создания меню, кнопочных форм, панелей инструментов |  |  |  |
|  | Реализация соответствующей иерархии классов, осуществление работы с информацией |  |  |  |
|  | Создание справочной системы |  |  |  |
|  | Реализация сохранения данных в файлы формата \*.docx, \*.txt, \*.bin, \*.rtf и чтение этих данных из них |  |  |  |

3.5 Описание справочной системы

Средством разработки для создания справочной системы является программное средство Dr.Explain, которое помогает создавать файлы справки, документацию и руководства пользователя [7].

Справочная система состоит из начальной страницы, оглавления, 9-ти разделов и 3-х подразделов.

Раздел 1 «Запуск приложения» Предоставляет описание процесса запуска приложения в системе.

Подраздел 1.1 «Шаги по установке Python» является инструкцией по установке Python, для работы скрипта программного средства.

Раздел 2 «Авторизация» Является подробным обзором страницы входа.

Раздел 3 «Регистрация» Предоставляет инструкцию по регистрации.

Подраздел 3.1 «Заполнение профиля» Предоставляет подробную инструкцию по начальному заполнению профиля и личной информации. Подраздел 3.2 «Начальная настройка» Предоставляет инструкцию по ранней настройке внешнего вида приложения.

Раздел 4 «Главная страница» Предоставляет информацию об элементах на главной странице программного средства.

Раздел 5 «Книга» Предоставляет инструкцию по использованию функций книги.

Раздел 6 «Задания» Предоставляет подробную информацию о странице задач и о самих задачах.

Раздел 7 «Профиль» Предоставляет пользователю информацию о его личных данных и профиле, а также возможности редактирования этих данных.

Раздел 8 «Статистика» Обозревает функции страницы статистики.

Раздел 9 «Страница настроек» Явно показывает функционал и советы по настройке.

Справочная система вызывается c помощью нажатия клавиши F1 на клавиатуре.

4 Применение

4.1 Назначение программного средства

Программное средство «PythonWave» предназначено для изучения языка программирования Python. Пользователи могут использовать это программное средство для обучения, создание отчётов своей продуктивности и получение базы Python. Так же познакомившись с этим языком, пользователь может понять симпатизирует ли ему этот язык в целом.

Ограничение, накладываемое на представленное программное средство, включает следующее: необходимость наличия операционной системы Windows.

4.2 Условия применения

Для функционирования программы требуется операционная система Windows, интерпретатор Python (рекомендуемые версии 3.12.3 или 3.12.4). Процесс запуска приложения заключается в перемещения содержимого папки «PythonWave» в выбранную директорию на компьютере и запуске файла «PythonWave.exe».

Заключение

Целью курсового проектирования было создание программного средства с целью обучения пользователя. Оно позволяет выполнять практические задачи, изучать теоретический материал, создавать отчеты, настраивать приложение. Была поставлена задача реализовать данное программное средство в среде разработки Visual Studio, на языке программирования C++ с использованием фреймворка CLI .NET. В процессе разработки серьезных трудностей не возникало.

Достоинством программного средства является простое взаимодействие пользователя с приложением. Удобный и лаконичный интерфейс с анимациями делает работу с программой максимально эффективной и приятной. Пользователю не нужно тратить много времени на изучение сложных инструкций или освоение новых навыков — все действия выполняются легко и быстро. Программа также обеспечивает высокий уровень интерактивности, что способствует лучшему усвоению материала.

Недостатком является то, что приложение является локальным. Пользователь не сможет войти в свой аккаунт на других устройствах, где нет сохраненных файлов. Это ограничивает мобильность и гибкость использования программного средства.

Улучшение данного программного средства возможно путем полного перехода на язык программирования C#, так как на нем корректно работают сторонние фреймворки. Подключение базы данных SQL для сохранения данных пользователя онлайн позволит создать систему учета прогресса, доступную с любого устройства. Также стоит рассмотреть доработку и пополнение задач для повышения их разнообразия и уровня сложности, что сделает обучение более комплексным и полезным.

Дополнительно можно рассмотреть внедрение возможностей для совместного обучения, таких как обмен задачами между пользователями и поддержка учебных групп, что позволит создать более интерактивное и социальное учебное окружение. Интеграция с популярными облачными сервисами для хранения данных также может повысить удобство использования приложения.

Список использованных источников

1. Багласова, Т.Г. Методические указания по оформлению курсовых и дипломных проектов / Т.Г. Багласова, К.О. Якимович. – Минск : КБП, 2013. – 29 c.
2. Блок-схема [Электронный ресурс]. – Wikipedia, 2024. – Режим доступа : https://ru.wikipedia.org/wiki/Блок-схема. – Дата доступа : 15.04.2024.
3. Диаграмма классов [Электронный ресурс]. – Wikipedia, 2024. – Режим доступа : https://ru.wikipedia.org/wiki/Диаграмма\_классов. – Дата доступа : 17.04.2024.
4. C++ [Электронный ресурс]. – Wikipedia, 2024. - Режим доступа : https://ru.wikipedia.org/wiki/C++. – Дата доступа : 11.05.2024.
5. Visual Studio 2022 [Электронный ресурс]. – Microsoft, 2024. – Режим доступа : https://visualstudio.microsoft.com/ru/vs. – Дата доступа : 13.05.2024.
6. Windows Forms [Электронный ресурс]. – Wikipedia, 2024. – Режим доступа: https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/overview/. Дата доступа : 15.05.2024.
7. Dr.Explain [Электронный ресурс]. – drexplain.ru, 2024. – Режим доступа: https://www.drexplain.ru/. – Дата доступа : 25.05.2024.

Приложение А

**(обязательное)**

**Текст программных модулей**

«auth.cpp»

#include "auth.h"

#include <Windows.h>

using namespace kurs;

[STAThreadAttribute]

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

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Application::Run(gcnew auth);

return 0;

}

«labs.cpp»

#include "labs.h"

«auth.h»

#pragma once

#include "labs.h"

namespace kurs {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

/// <summary>

/// Сводка для auth

/// </summary>

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

{

public:

auth(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~auth()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::ComboBox^ comboBox1;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::PictureBox^ pictureBox1;

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(auth::typeid));

this->comboBox1 = (gcnew System::Windows::Forms::ComboBox());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->button1 = (gcnew System::Windows::Forms::Button());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->BeginInit();

this->SuspendLayout();

//

// comboBox1

//

this->comboBox1->DropDownStyle = System::Windows::Forms::ComboBoxStyle::DropDownList;

this->comboBox1->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 10, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->comboBox1->FormattingEnabled = true;

this->comboBox1->Items->AddRange(gcnew cli::array< System::Object^ >(3) { L"Т-444", L"Т-445", L"Т-446" });

this->comboBox1->Location = System::Drawing::Point(118, 307);

this->comboBox1->Name = L"comboBox1";

this->comboBox1->Size = System::Drawing::Size(121, 24);

this->comboBox1->TabIndex = 0;

//

// textBox1

//

this->textBox1->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->textBox1->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->textBox1->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox1->ForeColor = System::Drawing::SystemColors::Desktop;

this->textBox1->Location = System::Drawing::Point(118, 273);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(121, 26);

this->textBox1->TabIndex = 1;

//

// textBox2

//

this->textBox2->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->textBox2->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->textBox2->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox2->ForeColor = System::Drawing::SystemColors::Desktop;

this->textBox2->Location = System::Drawing::Point(118, 340);

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(121, 26);

this->textBox2->TabIndex = 2;

//

// button1

//

this->button1->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->button1->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 14, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button1->Location = System::Drawing::Point(133, 386);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(87, 40);

this->button1->TabIndex = 3;

this->button1->Text = L"ВОЙТИ";

this->button1->UseVisualStyleBackColor = false;

this->button1->Click += gcnew System::EventHandler(this, &auth::button1\_Click);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label1->Location = System::Drawing::Point(55, 272);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(58, 19);

this->label1->TabIndex = 4;

this->label1->Text = L"Логин:";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label2->Location = System::Drawing::Point(51, 306);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(64, 19);

this->label2->TabIndex = 5;

this->label2->Text = L"Группа:";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label3->Location = System::Drawing::Point(45, 340);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(68, 19);

this->label3->TabIndex = 6;

this->label3->Text = L"Пароль:";

//

// pictureBox1

//

this->pictureBox1->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"pictureBox1.BackgroundImage")));

this->pictureBox1->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Zoom;

this->pictureBox1->Location = System::Drawing::Point(81, 24);

this->pictureBox1->Name = L"pictureBox1";

this->pictureBox1->Size = System::Drawing::Size(203, 211);

this->pictureBox1->TabIndex = 7;

this->pictureBox1->TabStop = false;

//

// auth

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::Color::LightSkyBlue;

this->ClientSize = System::Drawing::Size(358, 489);

this->Controls->Add(this->pictureBox1);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->button1);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->comboBox1);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedDialog;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"auth";

this->Text = L"163umneck";

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

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

try {

// Сбор логина и пароля с txt

String^ name = "login.txt";

StreamReader^ file = File::OpenText(name);

String^ log = file->ReadToEnd();

String^ l = Convert::ToString(this->textBox1->Text);

String^ p = Convert::ToString(this->textBox2->Text);

String^ namep = "password.txt";

StreamReader^ ffile = File::OpenText(namep);

String^ pas = ffile->ReadToEnd();

// Проверка введённых данных

if (log == l && p == pas && comboBox1->Text == "Т-444") {

StreamWriter^ group = gcnew StreamWriter("current.txt");

group->WriteLine("Т-444");

group->Close();

MessageBox::Show("Вы успешно вошли в аккаунт!");

labs^ form = gcnew labs;

this->Hide();

form->Show();

}

// Проверка введённых данных

else if (log == l && p == pas && comboBox1->Text == "Т-445") {

StreamWriter^ group = gcnew StreamWriter("current.txt");

group->WriteLine("Т-445");

group->Close();

MessageBox::Show("Вы успешно вошли в аккаунт!");

labs^ form = gcnew labs;

this->Hide();

form->Show();

}

// Проверка введённых данных

else if (log == l && p == pas && comboBox1->Text == "Т-446") {

StreamWriter^ group = gcnew StreamWriter("current.txt");

group->WriteLine("Т-446");

group->Close();

MessageBox::Show("Вы успешно вошли в аккаунт!");

labs^ form = gcnew labs;

this->Hide();

form->Show();

}

else {

MessageBox::Show("Проверьте введенные данные!");

this->textBox1->Clear();

this->comboBox1->Text = "";

this->textBox2->Clear();

}

}

catch (FormatException^ e) {

MessageBox::Show("Ошибка при авторизации");

}

}

};

}

«labs.h»

#pragma once

#include <cliext/map>

namespace kurs {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Collections::Generic;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

/// <summary>

/// Сводка для labs

/// </summary>

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

{

public:

labs(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~labs()

{

if (components)

{

delete components;

}

}

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container^ components;

private: System::Windows::Forms::DataGridView^ dataGridView1;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Button^ button3;

private: System::Windows::Forms::Button^ button4;

private: System::Windows::Forms::TextBox^ textBox4;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Button^ button2;

private: System::Windows::Forms::TextBox^ textBox5;

private: System::Windows::Forms::TextBox^ textBox6;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::ComboBox^ comboBox1;

private: System::Windows::Forms::ComboBox^ comboBox2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label7;

private: System::Windows::Forms::Button^ button5;

private: System::Windows::Forms::TextBox^ textBox3;

private: System::Windows::Forms::Button^ button6;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::Button^ button7;

private: System::Windows::Forms::Label^ label6;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(labs::typeid));

this->dataGridView1 = (gcnew System::Windows::Forms::DataGridView());

this->button1 = (gcnew System::Windows::Forms::Button());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->button3 = (gcnew System::Windows::Forms::Button());

this->button4 = (gcnew System::Windows::Forms::Button());

this->textBox4 = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->button2 = (gcnew System::Windows::Forms::Button());

this->textBox5 = (gcnew System::Windows::Forms::TextBox());

this->textBox6 = (gcnew System::Windows::Forms::TextBox());

this->label5 = (gcnew System::Windows::Forms::Label());

this->label6 = (gcnew System::Windows::Forms::Label());

this->comboBox1 = (gcnew System::Windows::Forms::ComboBox());

this->comboBox2 = (gcnew System::Windows::Forms::ComboBox());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label7 = (gcnew System::Windows::Forms::Label());

this->button5 = (gcnew System::Windows::Forms::Button());

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->button6 = (gcnew System::Windows::Forms::Button());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->button7 = (gcnew System::Windows::Forms::Button());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->BeginInit();

this->SuspendLayout();

//

// dataGridView1

//

this->dataGridView1->AllowUserToAddRows = false;

this->dataGridView1->AllowUserToDeleteRows = false;

this->dataGridView1->BackgroundColor = System::Drawing::SystemColors::ActiveCaption;

this->dataGridView1->ColumnHeadersBorderStyle = System::Windows::Forms::DataGridViewHeaderBorderStyle::Single;

this->dataGridView1->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->dataGridView1->Location = System::Drawing::Point(24, 23);

this->dataGridView1->Name = L"dataGridView1";

this->dataGridView1->RowHeadersWidth = 51;

this->dataGridView1->Size = System::Drawing::Size(406, 171);

this->dataGridView1->TabIndex = 0;

//

// button1

//

this->button1->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->button1->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button1->Location = System::Drawing::Point(596, 120);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(115, 31);

this->button1->TabIndex = 1;

this->button1->Text = L"Добавить ЛР";

this->button1->UseVisualStyleBackColor = false;

this->button1->Click += gcnew System::EventHandler(this, &labs::button1\_Click);

//

// textBox1

//

this->textBox1->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->textBox1->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->textBox1->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox1->ForeColor = System::Drawing::SystemColors::Desktop;

this->textBox1->Location = System::Drawing::Point(611, 40);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(100, 26);

this->textBox1->TabIndex = 2;

//

// textBox2

//

this->textBox2->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->textBox2->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->textBox2->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox2->ForeColor = System::Drawing::SystemColors::Desktop;

this->textBox2->Location = System::Drawing::Point(611, 84);

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(100, 26);

this->textBox2->TabIndex = 3;

//

// button3

//

this->button3->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->button3->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 14, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button3->Location = System::Drawing::Point(312, 209);

this->button3->Name = L"button3";

this->button3->Size = System::Drawing::Size(118, 32);

this->button3->TabIndex = 6;

this->button3->Text = L"Сохранить";

this->button3->UseVisualStyleBackColor = false;

this->button3->Click += gcnew System::EventHandler(this, &labs::button3\_Click);

//

// button4

//

this->button4->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->button4->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button4->Location = System::Drawing::Point(776, 121);

this->button4->Name = L"button4";

this->button4->Size = System::Drawing::Size(100, 31);

this->button4->TabIndex = 7;

this->button4->Text = L"Удалить ЛР";

this->button4->UseVisualStyleBackColor = false;

this->button4->Click += gcnew System::EventHandler(this, &labs::button4\_Click);

//

// textBox4

//

this->textBox4->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->textBox4->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->textBox4->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox4->ForeColor = System::Drawing::SystemColors::Desktop;

this->textBox4->Location = System::Drawing::Point(776, 84);

this->textBox4->Name = L"textBox4";

this->textBox4->Size = System::Drawing::Size(100, 26);

this->textBox4->TabIndex = 8;

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label1->Location = System::Drawing::Point(517, 42);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(77, 18);

this->label1->TabIndex = 9;

this->label1->Text = L"Номер ЛР:";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label2->Location = System::Drawing::Point(473, 86);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(121, 18);

this->label2->TabIndex = 10;

this->label2->Text = L"Дата начала ЛР:\r\n";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label4->Location = System::Drawing::Point(787, 59);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(75, 18);

this->label4->TabIndex = 12;

this->label4->Text = L"Номер ЛР";

//

// button2

//

this->button2->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->button2->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button2->Location = System::Drawing::Point(302, 347);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(83, 59);

this->button2->TabIndex = 13;

this->button2->Text = L"Экспорт отчета";

this->button2->UseVisualStyleBackColor = false;

this->button2->Click += gcnew System::EventHandler(this, &labs::button2\_Click\_1);

//

// textBox5

//

this->textBox5->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->textBox5->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->textBox5->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox5->ForeColor = System::Drawing::SystemColors::Desktop;

this->textBox5->Location = System::Drawing::Point(31, 360);

this->textBox5->Name = L"textBox5";

this->textBox5->Size = System::Drawing::Size(100, 26);

this->textBox5->TabIndex = 14;

//

// textBox6

//

this->textBox6->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->textBox6->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->textBox6->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox6->ForeColor = System::Drawing::SystemColors::Desktop;

this->textBox6->Location = System::Drawing::Point(183, 360);

this->textBox6->Name = L"textBox6";

this->textBox6->Size = System::Drawing::Size(100, 26);

this->textBox6->TabIndex = 15;

//

// label5

//

this->label5->AutoSize = true;

this->label5->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label5->Location = System::Drawing::Point(11, 364);

this->label5->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(15, 18);

this->label5->TabIndex = 16;

this->label5->Text = L"с";

//

// label6

//

this->label6->AutoSize = true;

this->label6->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label6->Location = System::Drawing::Point(153, 364);

this->label6->Margin = System::Windows::Forms::Padding(2, 0, 2, 0);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(25, 18);

this->label6->TabIndex = 17;

this->label6->Text = L"по";

//

// comboBox1

//

this->comboBox1->DropDownStyle = System::Windows::Forms::ComboBoxStyle::DropDownList;

this->comboBox1->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 10, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->comboBox1->FormattingEnabled = true;

this->comboBox1->Items->AddRange(gcnew cli::array< System::Object^ >(2) { L"лабораторные работы", L"фамилии" });

this->comboBox1->Location = System::Drawing::Point(135, 214);

this->comboBox1->Name = L"comboBox1";

this->comboBox1->Size = System::Drawing::Size(160, 24);

this->comboBox1->TabIndex = 18;

//

// comboBox2

//

this->comboBox2->DropDownStyle = System::Windows::Forms::ComboBoxStyle::DropDownList;

this->comboBox2->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 10, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->comboBox2->FormattingEnabled = true;

this->comboBox2->Items->AddRange(gcnew cli::array< System::Object^ >(2) { L"возрастанию", L"убыванию" });

this->comboBox2->Location = System::Drawing::Point(135, 254);

this->comboBox2->Name = L"comboBox2";

this->comboBox2->Size = System::Drawing::Size(160, 24);

this->comboBox2->TabIndex = 19;

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label3->Location = System::Drawing::Point(23, 213);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(99, 18);

this->label3->TabIndex = 20;

this->label3->Text = L"Сортировать:";

//

// label7

//

this->label7->AutoSize = true;

this->label7->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label7->Location = System::Drawing::Point(97, 254);

this->label7->Name = L"label7";

this->label7->Size = System::Drawing::Size(25, 18);

this->label7->TabIndex = 21;

this->label7->Text = L"по";

//

// button5

//

this->button5->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->button5->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button5->Location = System::Drawing::Point(149, 292);

this->button5->Name = L"button5";

this->button5->Size = System::Drawing::Size(118, 35);

this->button5->TabIndex = 22;

this->button5->Text = L"Сортировать";

this->button5->UseVisualStyleBackColor = false;

this->button5->Click += gcnew System::EventHandler(this, &labs::button5\_Click);

//

// textBox3

//

this->textBox3->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->textBox3->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->textBox3->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox3->ForeColor = System::Drawing::SystemColors::Desktop;

this->textBox3->Location = System::Drawing::Point(499, 219);

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(137, 26);

this->textBox3->TabIndex = 23;

//

// button6

//

this->button6->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->button6->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button6->Location = System::Drawing::Point(530, 254);

this->button6->Name = L"button6";

this->button6->Size = System::Drawing::Size(78, 41);

this->button6->TabIndex = 24;

this->button6->Text = L"Искать";

this->button6->UseVisualStyleBackColor = false;

this->button6->Click += gcnew System::EventHandler(this, &labs::button6\_Click);

//

// richTextBox1

//

this->richTextBox1->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->richTextBox1->Font = (gcnew System::Drawing::Font(L"Bahnschrift", 9, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->richTextBox1->Location = System::Drawing::Point(667, 179);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->ReadOnly = true;

this->richTextBox1->Size = System::Drawing::Size(209, 140);

this->richTextBox1->TabIndex = 25;

this->richTextBox1->Text = L"";

//

// button7

//

this->button7->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"button7.BackgroundImage")));

this->button7->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Zoom;

this->button7->Location = System::Drawing::Point(806, 338);

this->button7->Name = L"button7";

this->button7->Size = System::Drawing::Size(70, 68);

this->button7->TabIndex = 26;

this->button7->UseVisualStyleBackColor = true;

this->button7->Click += gcnew System::EventHandler(this, &labs::button7\_Click);

//

// labs

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::Color::LightSkyBlue;

this->ClientSize = System::Drawing::Size(905, 426);

this->Controls->Add(this->button7);

this->Controls->Add(this->richTextBox1);

this->Controls->Add(this->button6);

this->Controls->Add(this->textBox3);

this->Controls->Add(this->button5);

this->Controls->Add(this->label7);

this->Controls->Add(this->label3);

this->Controls->Add(this->comboBox2);

this->Controls->Add(this->comboBox1);

this->Controls->Add(this->label6);

this->Controls->Add(this->label5);

this->Controls->Add(this->textBox6);

this->Controls->Add(this->textBox5);

this->Controls->Add(this->button2);

this->Controls->Add(this->label4);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->textBox4);

this->Controls->Add(this->button4);

this->Controls->Add(this->button3);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->button1);

this->Controls->Add(this->dataGridView1);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedDialog;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->KeyPreview = true;

this->Name = L"labs";

this->Text = L"163umneck";

this->Load += gcnew System::EventHandler(this, &labs::labs\_Load);

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

private :

// Функция для проверки даты и окрашивания ячеек

void CheckDatesAndColorCells(System::Windows::Forms::DataGridView^ dataGridView) {

// Получите текущую дату

System::DateTime currentDate = System::DateTime::Now.Date;

// Просмотрите все столбцы

for (int i = 0; i < dataGridView->Columns->Count; ++i) {

// Получите название столбца

System::String^ columnName = dataGridView->Columns[i]->Name;

// Проверьте, содержит ли название столбца дату последней сдачи

if (columnName->Contains("(") && columnName->Contains(")")) {

// Извлеките дату последней сдачи из названия столбца

int startIndex = columnName->IndexOf("(") + 1;

int endIndex = columnName->IndexOf(")");

System::String^ endDateStr = columnName->Substring(startIndex, endIndex - startIndex);

// Преобразуйте строку в дату

System::DateTime endDate = System::DateTime::Parse(endDateStr);

// Если текущая дата превышает дату последней сдачи, окрасьте ячейки в красный цвет

// Если текущая дата равна дате последней сдачи, окрасьте ячейки в желтый цвет

// Если дата последней сдачи еще не наступила, окрасьте ячейки в белый цвет

for each (System::Windows::Forms::DataGridViewRow ^ row in dataGridView->Rows) {

// Проверьте, является ли ячейка пустой

if (!row->IsNewRow && (row->Cells[i]->Value == "" || row->Cells[i]->Value == nullptr)) {

if (currentDate > endDate) {

row->Cells[i]->Style->BackColor = System::Drawing::Color::Red;

}

else if (currentDate == endDate) {

row->Cells[i]->Style->BackColor = System::Drawing::Color::Yellow;

}

else if (currentDate < endDate) {

row->Cells[i]->Style->BackColor = System::Drawing::Color::White;

}

}

}

}

}

}

// Функция для сохранения данных из DataGridView в файл CSV

void SaveDataToCSV(System::Windows::Forms::DataGridView^ dataGridView, String^ filePath) {

StreamWriter^ sw = gcnew StreamWriter(filePath, false, System::Text::Encoding::Default);

String^ data = "";

// Запись заголовков столбцов

for each (DataGridViewColumn ^ column in dataGridView->Columns) {

data += column->HeaderText + ";";

}

sw->WriteLine(data);

// Запись данных

for each (DataGridViewRow ^ row in dataGridView->Rows) {

data = "";

for each (DataGridViewCell ^ cell in row->Cells) {

data += cell->Value + ";";

}

sw->WriteLine(data);

}

sw->Close();

}

// Функция для загрузки данных из файла CSV в DataGridView

void LoadDataFromCSV(System::Windows::Forms::DataGridView^ dataGridView, String^ filePath) {

StreamReader^ sr = gcnew StreamReader(filePath, System::Text::Encoding::Default);

String^ line;

// Чтение данных

line = sr->ReadLine();

if (line != nullptr) {

// Создание столбцов

array<String^>^ headers = line->Split(';');

for each (String ^ header in headers) {

dataGridView->Columns->Add(header, header);

}

// Чтение и добавление строк

while ((line = sr->ReadLine()) != nullptr) {

array<String^>^ items = line->Split(';');

dataGridView->Rows->Add(items);

}

for (int i = 0; i < dataGridView1->Columns->Count; i++)

{

if (String::IsNullOrEmpty(dataGridView1->Columns[i]->HeaderText))

{

dataGridView1->Columns->RemoveAt(i);

i--; // Уменьшаем индекс, т.к. столбец был удален

}

}

}

sr->Close();

}

#pragma endregion

private: System::Void labs\_Load(System::Object^ sender, System::EventArgs^ e) {

StreamReader^ group = File::OpenText("current.txt");

String^ name = group->ReadToEnd();

group->Close();

name = name->Trim();

String^ txt = Convert::ToString(name) + ".csv";

LoadDataFromCSV(dataGridView1, txt);

CheckDatesAndColorCells(dataGridView1);

dataGridView1->CellValueChanged += gcnew System::Windows::Forms::DataGridViewCellEventHandler(this, &labs::dataGridView1\_CellValueChanged);

FormClosing += gcnew System::Windows::Forms::FormClosingEventHandler(this, &labs::labs\_FormClosing);

ToolTip^ toolTip1 = gcnew ToolTip;

toolTip1->SetToolTip(this->textBox1, "В это поле можно вводить целые положительные числа");

ToolTip^ toolTip2 = gcnew ToolTip;

toolTip2->SetToolTip(this->textBox2, "В это поле вводится дата в формате день.месяц");

ToolTip^ toolTip3 = gcnew ToolTip;

toolTip3->SetToolTip(this->textBox3, "В это поле можно вводить фамилии учащихся для поиска по фамилии\r\nи 'ЛР'/'Лабораторная работа ' номер для поиска по ЛР");

ToolTip^ toolTip4 = gcnew ToolTip;

toolTip4->SetToolTip(this->textBox4, "В это поле можно вводить целые положительные числа");

ToolTip^ toolTip5 = gcnew ToolTip;

toolTip5->SetToolTip(this->textBox5, "В это поле вводится дата в формате день.месяц");

ToolTip^ toolTip6 = gcnew ToolTip;

toolTip6->SetToolTip(this->textBox6, "В это поле вводится дата в формате день.месяц");

}

private: System::Void labs\_FormClosing(System::Object^ sender, System::Windows::Forms::FormClosingEventArgs^ e) {

Application::Exit();

}

private: System::Void dataGridView1\_CellValueChanged(System::Object^ sender, System::Windows::Forms::DataGridViewCellEventArgs^ e) {

// Обработчик событий для изменения значений ячеек

System::Windows::Forms::DataGridView^ dataGridView = safe\_cast<System::Windows::Forms::DataGridView^>(sender);

if (e->RowIndex >= 0 && e->ColumnIndex >= 0) {

System::Windows::Forms::DataGridViewCell^ cell = dataGridView->Rows[e->RowIndex]->Cells[e->ColumnIndex];

if (cell->Value != nullptr && cell->Value->ToString() != "") {

try {

int value = Convert::ToInt32(cell->Value);

if (value >= 1 && value <= 10) {

cell->Tag = cell->Value; // Сохраняем новое значение в Tag

}

else {

// Недопустимое значение, возвращаем старое и выводим предупреждение

System::Windows::Forms::MessageBox::Show("Пожалуйста, введите целое число от 1 до 10.", "Неверное значение", System::Windows::Forms::MessageBoxButtons::OK, System::Windows::Forms::MessageBoxIcon::Warning);

cell->Value = cell->Tag != nullptr ? cell->Tag->ToString() : "";

}

}

catch (FormatException^) {

// Недопустимое значение, возвращаем старое и выводим предупреждение

System::Windows::Forms::MessageBox::Show("Пожалуйста, введите целое число от 1 до 10.", "Неверное значение", System::Windows::Forms::MessageBoxButtons::OK, System::Windows::Forms::MessageBoxIcon::Warning);

cell->Value = cell->Tag != nullptr ? cell->Tag->ToString() : "";

}

}

else {

// Если значение пустое, сохраняем его в Tag и не показываем предупреждение

cell->Tag = "";

}

// Если значение пустое, сохраняем пустое значение в Tag

if (cell->Value == nullptr || cell->Value->ToString() == "") {

cell->Tag = "";

}

// Если ячейка была красной и пустой, то поменять её цвет на белый при изменении значения

if (cell->Style->BackColor == System::Drawing::Color::Red && cell->Value != "" && cell->Value != nullptr) {

cell->Style->BackColor = System::Drawing::Color::White;

}

}

// Применяем проверку дат и окраску ячеек

CheckDatesAndColorCells(dataGridView);

}

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

try {

if (textBox1->Text != "") {

String^ labNumber = textBox1->Text;

System::Text::RegularExpressions::Regex^ integerRegex = gcnew System::Text::RegularExpressions::Regex("^[0-9]+$");

if (!integerRegex->IsMatch(labNumber)) {

MessageBox::Show("В поле 'Номер ЛР' можно вводить только целые числа");

textBox1->Clear();

return;

}

// Проверка на существование номера лабораторной работы

for (int i = 0; i < dataGridView1->Columns->Count; ++i) {

if (dataGridView1->Columns[i]->Name->Contains("ЛР" + labNumber)) {

MessageBox::Show("Такой номер лабораторной работы уже существует");

textBox1->Clear();

return;

; }

}

if (textBox2->Text != "") {

String^ startDate = textBox2->Text;

// Проверка формата даты

System::Text::RegularExpressions::Regex^ dateRegex = gcnew System::Text::RegularExpressions::Regex("^\\d{2}\\.\\d{2}$");

if (!dateRegex->IsMatch(startDate)) {

MessageBox::Show("Дата начала ЛР введена в неправильном формате. Формат должен быть 01.01 (число.месяц)");

textBox2->Clear();

return;

}

// Расчет конечной даты

array<String^>^ dateParts = startDate->Split('.');

int day = Int32::Parse(dateParts[0]);

int month = Int32::Parse(dateParts[1]);

DateTime startDateTime = DateTime(DateTime::Now.Year, month, day);

DateTime endDateTime = startDateTime.AddDays(14);

String^ endDate = endDateTime.ToString("dd.MM");

String^ columnName = "ЛР" + labNumber + " " + startDate + " (" + endDate + ")";

dataGridView1->Columns->Add(columnName, columnName);

// Заполните новый столбец пустыми значениями

for each (DataGridViewRow ^ row in dataGridView1->Rows) {

row->Cells[columnName]->Value = "";

}

CheckDatesAndColorCells(dataGridView1);

StreamReader^ group = File::OpenText("current.txt");

String^ name = group->ReadToEnd();

group->Close();

name = name->Trim();

String^ tex = Convert::ToString(name) + ".csv";

SaveDataToCSV(dataGridView1, tex);

}

else {

MessageBox::Show("Введите дату начала ЛР");

}

}

else {

MessageBox::Show("Введите номер лабораторной работы");

}

}

catch (Exception^ ex) {

MessageBox::Show("Введена некорректная дата");

textBox2->Clear();

}

}

private: System::Void button3\_Click(System::Object^ sender, System::EventArgs^ e) {

StreamReader^ group = File::OpenText("current.txt");

String^ name = group->ReadToEnd();

group->Close();

name = name->Trim();

String^ tex = Convert::ToString(name) + ".csv";

SaveDataToCSV(dataGridView1,tex);

CheckDatesAndColorCells(dataGridView1);

}

private: System::Void button4\_Click(System::Object^ sender, System::EventArgs^ e) {

if (textBox4->Text != "") {

String^ labNumber = textBox4->Text;

System::Text::RegularExpressions::Regex^ integerRegex = gcnew System::Text::RegularExpressions::Regex("^[0-9]+$");

if (!integerRegex->IsMatch(labNumber)) {

MessageBox::Show("В поле 'Номер ЛР' можно вводить только целые числа");

textBox4->Clear();

return;

}

if (textBox4->Text != "" && (Convert::ToInt32(textBox4->Text) > 0)) {

String^ labNumber = textBox4->Text;

int columnIndex = -1;

// Проверка на существование номера лабораторной работы

for (int i = 0; i < dataGridView1->Columns->Count; ++i) {

if (dataGridView1->Columns[i]->Name->Contains("ЛР" + labNumber)) {

columnIndex = i;

break;

}

}

if (columnIndex != -1) {

dataGridView1->Columns->RemoveAt(columnIndex);

}

else {

MessageBox::Show("Лабораторная работа с номером " + labNumber + " не найдена.");

}

}

else {

MessageBox::Show("Введите корректный номер лабораторной работы для удаления");

this->textBox4->Clear();

}

}

else {

MessageBox::Show("Введите номер лабораторной работы");

}

}

private: System::Void button2\_Click\_1(System::Object^ sender, System::EventArgs^ e) {

// Чтение дат из текстбоксов

if (textBox5->Text != "") {

if (textBox6->Text != "") {

String^ startDateStr = textBox5->Text;

String^ endDateStr = textBox6->Text;

// Проверка формата дат с использованием регулярных выражений

System::Text::RegularExpressions::Regex^ dateRegex = gcnew System::Text::RegularExpressions::Regex("^\\d{2}\\.\\d{2}$");

if (!dateRegex->IsMatch(startDateStr)) {

MessageBox::Show("Дата начала введена в неправильном формате. Формат должен быть 01.01 (число.месяц)");

textBox5->Clear();

return;

}

if (!dateRegex->IsMatch(endDateStr)) {

MessageBox::Show("Дата окончания введена в неправильном формате. Формат должен быть 01.01 (число.месяц)");

textBox6->Clear();

return;

}

// Преобразование строковых дат в DateTime, добавление текущего года для обработки даты

DateTime startDate = DateTime::ParseExact(startDateStr + ".2024", "dd.MM.yyyy", nullptr);

DateTime endDate = DateTime::ParseExact(endDateStr + ".2024", "dd.MM.yyyy", nullptr);

String^ report = "";

// Перебор всех столбцов, начиная со второго, поскольку первый столбец содержит фамилии

for (int colIndex = 1; colIndex < dataGridView1->Columns->Count; ++colIndex) {

String^ columnName = dataGridView1->Columns[colIndex]->Name;

// Извлечение и проверка даты окончания из названия столбца

if (columnName->Contains("ЛР") && columnName->Contains("(") && columnName->Contains(")")) {

int endDateIndex = columnName->IndexOf("(") + 1;

String^ endDateStrColumn = columnName->Substring(endDateIndex, columnName->IndexOf(")") - endDateIndex);

DateTime labEndDate = DateTime::ParseExact(endDateStrColumn + ".2024", "dd.MM.yyyy", nullptr);

// Проверка, находится ли дата окончания в заданном диапазоне

if (labEndDate >= startDate && labEndDate <= endDate) {

// Добавление абзацов для разделения лабораторных работ

if (!String::IsNullOrEmpty(report)) {

report += "\r\n";

}

report += columnName->Substring(0, columnName->IndexOf(" ")) + ":\r\n";

String^ notSubmitted = "Не сдали: ";

bool hasNonSubmitted = false;

bool hasSubmitted = false;

for each (System::Windows::Forms::DataGridViewRow ^ row in dataGridView1->Rows) {

if (row->IsNewRow) continue;

String^ studentName = row->Cells[0]->Value->ToString(); // Фамилия ученика

auto cellValue = row->Cells[colIndex]->Value; // Оценка за лабораторную работу

if (cellValue != nullptr && cellValue->ToString() != "") {

report += studentName + " " + cellValue->ToString() + ". ";

hasSubmitted = true;

}

else {

hasNonSubmitted = true;

notSubmitted += studentName + ", ";

}

}

// Удаление лишней запятой и пробела в конце строки для "Не сдали"

if (hasNonSubmitted) {

notSubmitted = notSubmitted->TrimEnd(gcnew array<wchar\_t> { ',', ' ' });

report += "\r\n" + notSubmitted;

}

report += "\r\n";

}

}

}

report = report->TrimEnd();

SaveFileDialog^ saveFileDialog1 = gcnew SaveFileDialog();

saveFileDialog1->Title = "Сохранить отчет";

saveFileDialog1->DefaultExt = "txt";

saveFileDialog1->Filter = "Text files (\*.txt)|\*.txt|All files (\*.\*)|\*.\*";

saveFileDialog1->RestoreDirectory = true;

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

StreamWriter^ sw = gcnew StreamWriter(saveFileDialog1->FileName, false, System::Text::Encoding::Default);

sw->WriteLine(report);

sw->Close();

System::Windows::Forms::MessageBox::Show("Отчет был создан", "", System::Windows::Forms::MessageBoxButtons::OK, System::Windows::Forms::MessageBoxIcon::Information);

}

}

else {

MessageBox::Show("Введите конечную дату");

}

}

else {

MessageBox::Show("Введите начальную дату");

}

}

// Класс компаратора для сортировки столбцов

ref class ColumnComparer : public System::Collections::Generic::IComparer<DataGridViewColumn^> {

public:

String^ sortOrder;

ColumnComparer(String^ sortOrder) {

this->sortOrder = sortOrder;

}

virtual int Compare(DataGridViewColumn^ a, DataGridViewColumn^ b) {

int aNumber = Int32::Parse(a->Name->Substring(2, a->Name->IndexOf(" ") - 2));

int bNumber = Int32::Parse(b->Name->Substring(2, b->Name->IndexOf(" ") - 2));

return sortOrder == "возрастанию" ? aNumber.CompareTo(bNumber) : bNumber.CompareTo(aNumber);

}

};

private: System::Void button5\_Click(System::Object^ sender, System::EventArgs^ e) {

// Получаем выбранные критерии сортировки

if (comboBox1->Text != "") {

if (comboBox2->Text != "") {

String^ sortField = comboBox1->SelectedItem->ToString();

String^ sortOrder = comboBox2->SelectedItem->ToString();

// Проверяем, какой критерий сортировки был выбран

if (sortField == "фамилии") {

// Сортируем по фамилии

dataGridView1->Sort(dataGridView1->Columns["Фамилия"], sortOrder == "возрастанию" ? ListSortDirection::Ascending : ListSortDirection::Descending);

}

else if (sortField == "лабораторные работы") {

// Сортируем столбцы по номеру лабораторной работы

List<DataGridViewColumn^>^ columns = gcnew List<DataGridViewColumn^>();

for (int i = 1; i < dataGridView1->Columns->Count; i++) {

DataGridViewColumn^ column = dataGridView1->Columns[i];

if (column->Name->StartsWith("ЛР")) {

columns->Add(column);

}

}

columns->Sort(gcnew ColumnComparer(sortOrder));

for (int i = 0; i < columns->Count; i++) {

columns[i]->DisplayIndex = i + 1;

}

}

}

else {

MessageBox::Show("Введите критерий сортировки");

}

}

else {

MessageBox::Show("Введите параметр сортировки");

}

}

private: System::Void button6\_Click(System::Object^ sender, System::EventArgs^ e) {

if (textBox3->Text != "") {

// Получаем введенное пользователем значение для поиска

String^ searchValue = textBox3->Text;

try {

// Очищаем richtextbox1 перед выводом результатов

richTextBox1->Clear();

if (searchValue->StartsWith("ЛР")) {

// Поиск по номеру лабораторной работы

String^ labNumberStr = searchValue->Substring(2)->Trim(); // Изменено здесь

int labNumber = Int32::Parse(labNumberStr); // Изменено здесь

String^ labColumn = "ЛР" + labNumber;

String^ passed = "Сдали: ";

String^ notPassed = "Не сдали: ";

bool found = false;

for (int i = 0; i < dataGridView1->Rows->Count; i++) {

DataGridViewRow^ row = dataGridView1->Rows[i];

for (int j = 1; j < dataGridView1->Columns->Count; j++) {

DataGridViewColumn^ column = dataGridView1->Columns[j];

if (column->Name->StartsWith(labColumn)) {

String^ cellValue = row->Cells[column->Name]->Value != nullptr ? row->Cells[column->Name]->Value->ToString() : "";

found = true;

if (!String::IsNullOrEmpty(cellValue)) {

passed += row->Cells["Фамилия"]->Value->ToString() + " (" + cellValue + "), ";

}

else {

notPassed += row->Cells["Фамилия"]->Value->ToString() + ", ";

}

}

}

}

if (found) {

richTextBox1->Text = "Лабораторная работа " + labNumber + ":\n" + passed->TrimEnd(',', ' ') + "\n" + notPassed->TrimEnd(',', ' ');

}

else {

MessageBox::Show("Не найдено результата по запросу " + searchValue);

}

}

else {

// Поиск по фамилии

String^ passed = "Сдал: ";

String^ notPassed = "Не сдал: ";

bool found = false;

for (int i = 0; i < dataGridView1->Rows->Count; i++) {

DataGridViewRow^ row = dataGridView1->Rows[i];

if (row->Cells["Фамилия"]->Value->ToString()->Equals(searchValue, StringComparison::OrdinalIgnoreCase)) {

found = true;

for (int j = 1; j < dataGridView1->Columns->Count; j++) {

DataGridViewCell^ cell = row->Cells[j];

String^ cellValue = cell->Value != nullptr ? cell->Value->ToString() : "";

String^ labWork = cell->OwningColumn->Name->Substring(0, cell->OwningColumn->Name->IndexOf(' '));

if (!String::IsNullOrEmpty(cellValue)) {

passed += labWork + " (" + cellValue + "), ";

}

else {

notPassed += labWork + ", ";

}

}

break;

}

}

if (found) {

richTextBox1->Text = "Учащийся " + searchValue + ":\n" + passed->TrimEnd(',', ' ') + "\n" + notPassed->TrimEnd(',', ' ');

}

else {

MessageBox::Show("Не найдено результата по запросу " + searchValue);

}

}

}

catch (Exception^ ex) {

// Обрабатываем исключения и выводим сообщение об ошибке

MessageBox::Show("Не найдено результата по запросу " + searchValue);

}

}

else {

MessageBox::Show("Введите запрос для поиска");

}

}

private: System::Void button7\_Click(System::Object^ sender, System::EventArgs^ e) {

String^ path = System::IO::Path::GetFullPath("Справочная система.pdf");

if (!System::IO::File::Exists(path)) return;

System::Diagnostics::Process::Start(path);

}

};

}

Приложение Б

**(справочное)**

**Результаты работы приложения**

На рисунках Б.1 - Б.10 представлены результаты тестирования приложения.

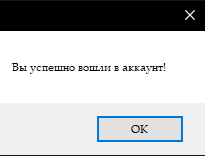


Рисунок Б.1 – Предупреждение об успешном входе

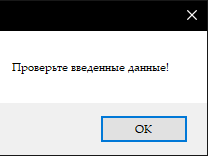


Рисунок Б.2 – Предупреждение об введении некорректных данных

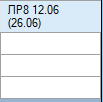


Рисунок Б.3 – Добавление лабораторной работы в таблицу

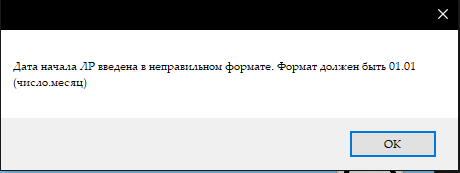


Рисунок Б.4 – Предупреждение об введении некорректных данных

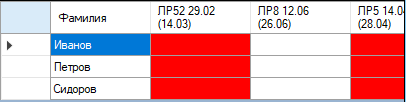


Рисунок Б.5 – Сортировка таблицы

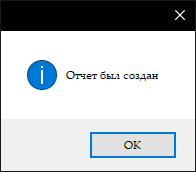


Рисунок Б.6 – Предупреждение об успешном создании отчета

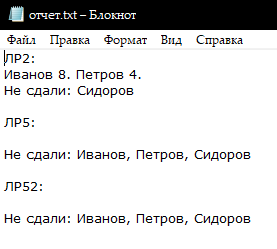


Рисунок Б.7 – Файл отчета

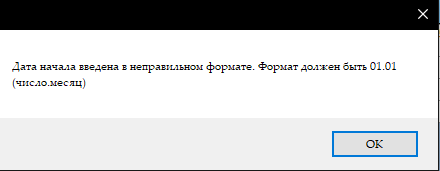


Рисунок Б.8 – Предупреждение об введении некорректных данных

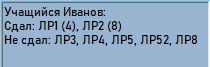


Рисунок Б.9 – Поиск данных из таблицы

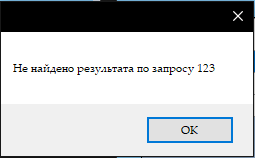


Рисунок Б.10 – Предупреждение об отсутствии результата