Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет инженерно-экономический

Кафедра экономической информатики

Дисциплина «Основы алгоритмизации и программирования»

|  |  |
| --- | --- |
|  | «К ЗАЩИТЕ ДОПУСТИТЬ» |
|  | Руководитель курсового проекта  Ассистент кафедры ЭИ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.А. Кириенко |
|  | \_\_\_.\_\_\_\_.2025 |

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовой работе

на тему:

**«****Программное средство организации выполнения курсовой работы студентами группы»**

БГУИР КР 1-40 05 01-02 025 ПЗ

|  |  |
| --- | --- |
|  | Выполнил студент группы 478101  Смертьев Владислав Валерьевич  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |
|  | Курсовая работа представлена на проверку \_\_\_.\_\_\_\_.2025  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |

Минск 2025

**РЕФЕРАТ**

БГУИР КР 11-40 05 01-02 025 ПЗ

**Смертьев В.В.** Программное средство организации выполнения курсовой работы студентами группы / В.В. Смертьев – Минск: БГУИР, 2025. – 43 с.

Пояснительная записка 43 с., 54 рис., 4 табл., 8 источников, 2 приложения

ПРОГРАММНОЕ СРЕДСТВО ОРГАНИЗАЦИИ ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ СТУДЕНТАМИ ГРУППЫ, модель *IDEF*0, схемы алгоритмов, программное средство

*Цель курсовой работы***:** разработка и анализ автоматизированной системы организации выполнения курсовых работ студентами группы, направленной на повышение эффективности учебного процесса и качества контроля выполнения заданий. Исследование охватывает оптимизацию процессов распределения тем, мониторинга выполнения этапов, взаимодействия между студентами и преподавателями, а также автоматизированное формирование отчетности, что способствует совершенствованию организации учебной деятельности [1].

*Методология проведения работы* основывалась на применении принципов системного подхода, использовании аналитических методов исследования учебных процессов, современных технологий обработки данных, а также методов моделирования организационных процессов в образовательной среде [1,4].

*Результаты работы* включают четкую постановку задачи и определение методов ее решения. Были разработаны модели учебного процесса с использованием нотации IDEF0, подробно описаны алгоритмы работы системы, составлено руководство пользователя. Проведенное тестирование подтвердило соответствие программного средства функциональным требованиям технического задания.

*Техническая реализация* выполнена на языке программирования C++ в среде разработки MS Visual Studio 2022. Особое внимание уделялось созданию удобного интерфейса и обеспечению надежности хранения данных [2,3,5].

*Область применения* разработанной системы охватывает весь процесс организации курсовых работ в учебных группах. Программное средство позволяет преподавателям эффективно распределять темы, контролировать сроки выполнения, анализировать прогресс студентов и формировать отчеты. Система полностью соответствует поставленным требованиям и существенно упрощает процесс организации и контроля выполнения курсовых работ.

**ОГЛАВЛЕНИЕ**

[**ВВЕДЕНИЕ** 4](#_Toc198125571)

[**1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ** 5](#_Toc198125572)

[**2** **ПРОЕКТИРОВАНИЕ И КОНСТРУИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА** 6](#_Toc198125573)

[**2.1** **Постановка задачи** 6](#_Toc198125574)

[**2.2** **Разработка функциональной модели предметной области в нотации IDEF0** 7](#_Toc198125575)

[**2.3** **Описание организации структур, хранящих данные.** 12](#_Toc198125576)

[**2.4** **Разработка перечня пользовательских функций программы.** 16](#_Toc198125577)

[**2.5** **Разработка схем алгоритмов работы программы** 17](#_Toc198125578)

[**3** **ТЕСТИРОВАНИЕ И ПРОВЕРКА РАБОТОСПОСОБНОСТИ ПРОГРАММНОГО СРЕДСТВА** 20](#_Toc198125579)

[**4 ИНСТРУКЦИЯ ПО РАЗВЁРТЫВАНИЮ ПРИЛОЖЕНИЯ И СКВОЗНОЙ ПРИМЕР** 23](#_Toc198125580)

[**4.1 Главное меню** 23](#_Toc198125581)

[**4.2 Модуль администратора** 24](#_Toc198125582)

[**4.3 Модуль пользователя** 34](#_Toc198125583)

[**Заключение** 37](#_Toc198125584)

[**Список использованных источников** 38](#_Toc198125585)

[**ПРИЛОЖЕНИЕ А (обязательное)** 38](#_Toc198125586)

[**Отчет о проверке на заимствование в системе «Антиплагиат»** 39](#_Toc198125587)

[**ПРИЛОЖЕНИЕ Б (обязательное)** 40](#_Toc198125588)

[**Листинг кода алгоритмов, реализующих основную бизнес-логику** 40](#_Toc198125589)

# **ВВЕДЕНИЕ**

Современный образовательный процесс в высших учебных заведениях предполагает активное вовлечение студентов в самостоятельную и проектную деятельность, ключевым элементом которой является выполнение курсовых работ. Однако организация этой деятельности, особенно в условиях групповой работы, сталкивается с рядом системных проблем: отсутствие четкой координации между участниками, трудности в отслеживании индивидуального прогресса, неравномерное распределение заданий, а также ограниченные возможности оперативной обратной связи с преподавателями. Эти факторы не только снижают эффективность учебного процесса, но и негативно сказываются на качестве итоговых работ.

Решением этой проблемы является создание компьютерных системы, способной сортировать и хранить информацию по мере ее поступления. Например, преподавателям и университету нужна программа для учета курсовых работы, которая сделает этот процесс быстрее и проще. Система должна быть легкой для понимания и использования любым пользователем.

Актуальность разработки специализированного программного средства для организации выполнения курсовых работ студентами в группе обусловлена необходимостью автоматизировать рутинные процессы, повысить прозрачность этапов проекта и оптимизировать коммуникацию между всеми участниками. Внедрение цифровых инструментов в образовательную среду соответствует тенденциям цифровизации образования, что подтверждается растущим интересом к LMS-платформам (Learning Management Systems) и другим инструментам для управления образовательными проектами

В ходе работы использовался комплекс методик, включающий анализ научной литературы, моделирование бизнес-процессов, а также методы тестирования для оценки удобства использования интерфейса.

Целью данной курсовой работы является разработка программного средства для организации и контроля выполнения курсовых работ студентами в группе.

Для достижения поставленной цели необходимо решить следующие задачи:

* Провести анализ текущей системы организации выполнения курсовых работ.
* Сформулировать требования к разрабатываемому программному средству.
* Спроектировать архитектуру системы.
* Оценить эффективность решения посредством тестирования.

# **1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ**

Организация выполнения курсовой работы - сложный процесс. Для его осуществления необходимо взаимодействие многих людей: студентов, преподавателей и администрации учебного заведения. Курсовая работа - важная часть обучения студента, она помогает ему развить навыки, необходимые для выбранной профессии. Традиционно этот процесс строится на ручном взаимодействии: студенты получают темы работ, самостоятельно планируют этапы выполнения, согласовывают промежуточные результаты с преподавателями, а преподаватели контролируют прогресс с помощью личных консультаций, переписки по электронной почте или обсуждений в общих чатах.

Одна из главных проблем нынешнего подхода - отсутствие общего пространства для управления проектами. Данные о прогрессе часто разрознены: ученики записывают задания в личные блокноты или цифровые заметки, учителя используют электронные таблицы для контроля сроков, а администрация создаёт отчеты в ручную на основе разрозненной информации. Это может привести к таким проблемам, как несогласованность, задержка обратной связи и дублирование информации. Например, учителя могут тратить много времени на сбор актуальных версий работ учеников, вместо того чтобы сосредоточиться на их содержательной оценке [6].

Еще одна проблема - неравномерное распределение нагрузки между студентами и преподавателями. Студенты, особенно на ранних этапах, не всегда обладают хорошими навыками распределения временных ресурсов, что приводит к появлению «завалов» по мере приближения сроков сдачи. Преподаватели, курирующие несколько групп, сталкиваются с необходимостью контролировать десятки заданий одновременно, что повышает риск упустить критические ошибки или нарушить сроки. Более того, без автоматизированных инструментов администрации сложно быстро создавать отчеты о прогрессе преподавателей или их нагрузке [7,8].

Разработанная программа должна позволять студенту получить информацию о сдаче и теме текущей курсовой работы, а так же предоставить доступ к её публикации от имени своей учетной записи. Для администрации и преподавателей программа должна иметь более широкий спектр возможностей: распределение тем среди студентов, заимствование доступа просмотра к новым опубликованным работам студентов и возможность их рецензирования.

Таким образом, предметная область исследования включает в себя процессы, методы и инструменты, связанные с планированием, выполнением и контролем курсовой работы в условиях групповой работы студентов. Ее специфика заключается в необходимости совмещения педагогических целей (формирование навыков самостоятельной работы, соблюдение академических требований) с организационными задачами (распределение ресурсов, минимизация рутинных операций) [6,7].

# **ПРОЕКТИРОВАНИЕ И КОНСТРУИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА**

## **Постановка задачи**

По условию задания требуется разработать программу организации выполнения студентами курсовых работ студентами. Для корректной работы система должна предусматривать режимы введения данных о клиенте, отражающее некоторые данные, связанные с его текущей курсовой работой. Среди них:

* Фамилия;
* Имя;
* Отчество;
* Группа студента;
* Курс обучения студента;
* Тема курсовой работы студента;
* Статус аттестации контрольных точек;
* Ссылка на ресурс с курсовой работой в режиме просмотра.

Предусмотрим следующие ограничения для программы:

* фамилия, имя и отчество студента необходимо указывать, используя буквы русского алфавита;
* Числа в дате сдач контрольных точек не должны выходить за логические рамки;
* Ссылка ресурса размещения курсовой работы работает корректно и отображает необходимые данные;

Разрабатываемая система должна обеспечивать разделение функционала по ролям: студент (пользователь), преподаватель (администратор). Обе группы должны иметь возможность зарегистрироваться и войти в систему.

Для студента предусмотрены следующие возможности:

* Просмотр личных данных, включая ФИО, группу, курс, тему курсовой работы, статус аттестации контрольных точек и ссылку на размещенную работу.
* Публикация курсовой работы через прикрепление корректной ссылки на ресурс (например, облачное хранилище).

Преподаватель получает расширенный функционал:

* Распределение тем курсовых работ среди студентов группы.
* Просмотр опубликованных работ через предоставленные ссылки с возможностью рецензирования.
* Контроль сроков сдачи контрольных точек их оценивание
* Создание/изменение дат контрольных точек (с учетом оговоренных ограничений).

Работа с учётными записями студентов выполняется только преподавателем и предполагает следующие функции:

* просмотреть все учётные записи;
* добавить новую учётную запись;
* редактировать учётную запись;
* удалить учётную запись.

Для облегчения работы с массивом данных в программе предусмотрены следующие функции:

* Поиск студентов по соответствующим параметрам;
* Сортировку клиентов по соответствующим параметрам;

Реализация всех вышеперечисленных модулей/подмодулей должно быть создано меню с соответствующими пунктами.

## **Разработка функциональной модели предметной области в нотации IDEF0**

Методология IDEF0 представляет собой стандартизированный подход к функциональному моделированию, который позволяет структурировать процессы проектирования и управления сложными системами. В рамках темы курсовой работы — разработки программного средства для организации выполнения курсовых работ студентами группы — методология IDEF0 применяется для анализа требований, проектирования архитектуры системы и формализации взаимодействия между её компонентами [3].

На первом этапе строится контекстная диаграмма, определяющая границы системы и её взаимодействие с внешней средой. В данном случае объектом моделирования выступает программное средство, представленное в виде единого блока

На рисунке 2.1 представлена основная диаграмма IDEF0. На изображении описан основной блок «Организовать выполнение курсовой работы студентами». Входящая стрелки — «Запрос студента», «Запрос преподавателя», «Темы курсовых работ». Это то, что необходимо иметь для начало работы. Управляющие стрелки – «Срок курсовой работы», «Нормативные документы» и «Критерии оценивания». В роли «Механизмов» выступают «Преподаватель», «Кафедра», «Студенты» и «Система контроля работ». После завершения процесса появляются «Заполненная база данных курсовых работ» и «Отчет о выполнении курсовых работ».



Рисунок 2.1 – Контекстная диаграмма информационной системы проверки курсовых работ

На рисунке 2.2 представлена диаграмма декомпозиции основного процесса. В данном случае процесс делится на три этапа: «Распределить темы среди студентов», «Записать работы студентов» и «Составить отчет».

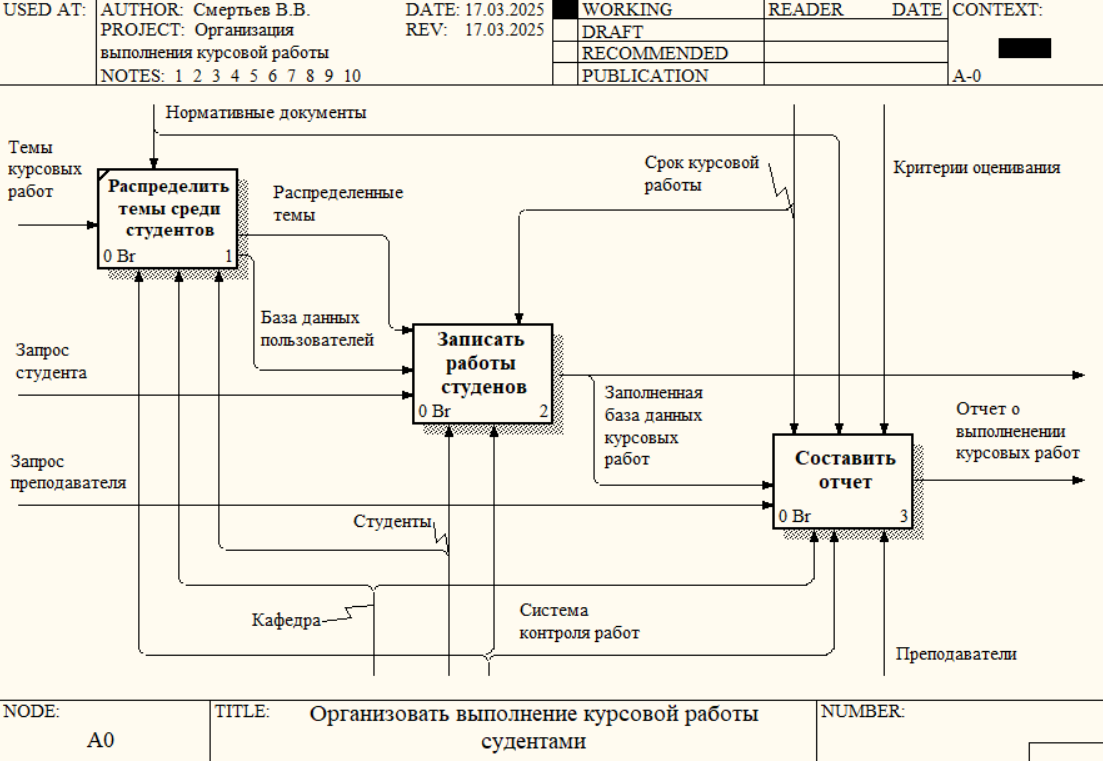


Рисунок 2.2 – Декомпозиция контекстной диаграммы

На рисунке 2.3 представлена декомпозиция процесса «Записать работы студента». В данном случае работа разбивается на три этапа: «Запросить логин/пароль», «Индефицировать студента», «Сохранить изменения студента».

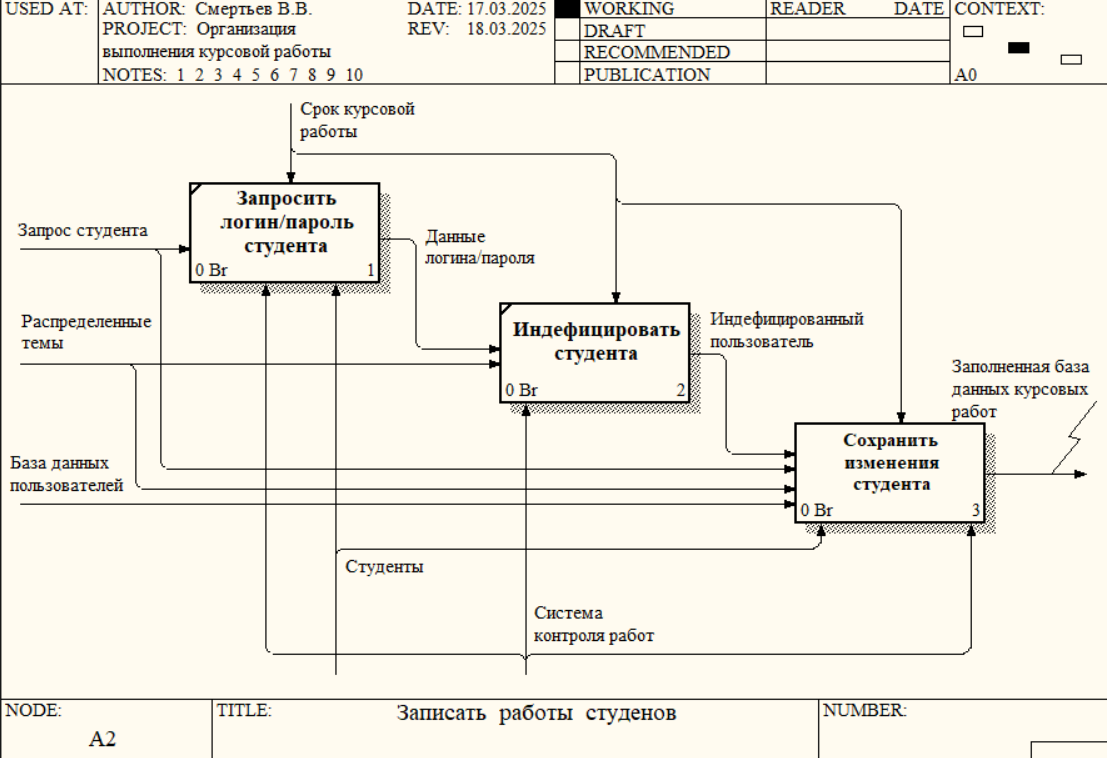


Рисунок 2.3 – Декомпозиция процесса «Записать работы студентов»

На рисунке 2.4 представлена декомпозиция процесса «Сохранить изменения студента». Этот процесс разбивается на «Определить тип запроса», «Внести изменения в базу данных», «Сообщить об изменении студенту».

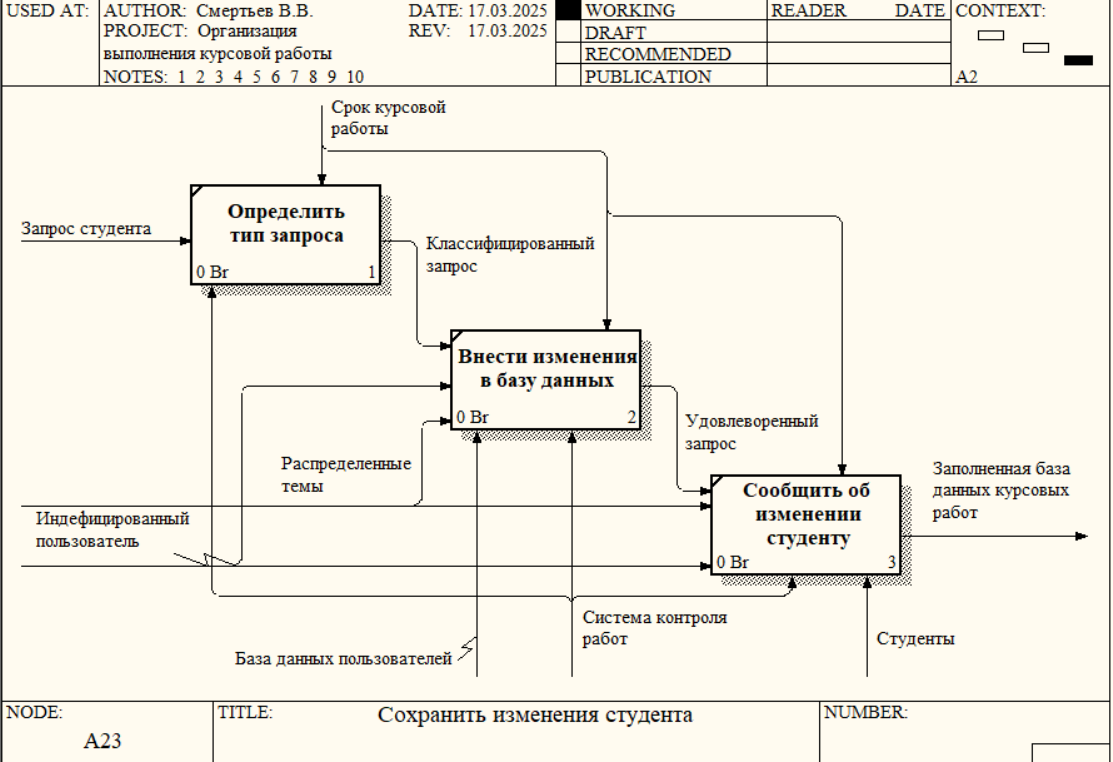


Рисунок 2.4 – Декомпозиция процесса «Сохранить изменения студента»

На рисунке 2.5 представлена декомпозиция процесса «Составить отчет». Этот процесс разделяется на три действия: «Запросить логин/пароль преподавателя», «Индефицировать преподавателся», «Завершить рецензирование работ».

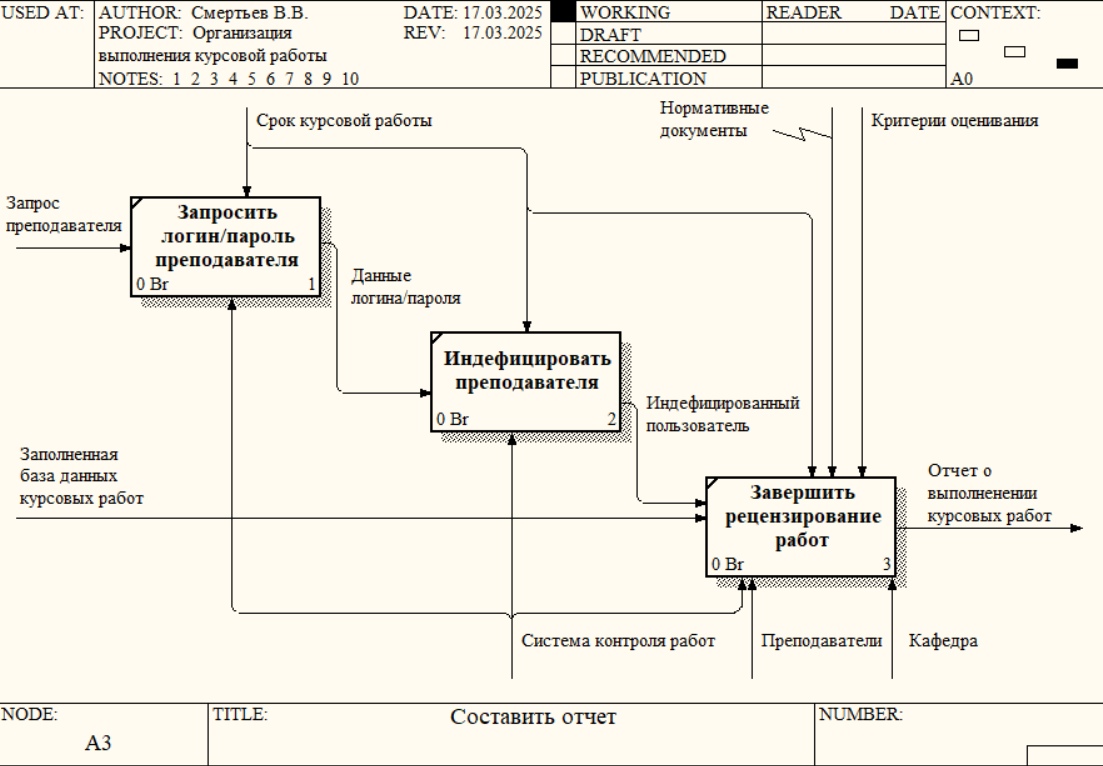


Рисунок 2.5 – Декомпозиция процесса «Составить отчет»

Данная модель представлена в нотации TO-BE, то есть содержит функционал процесса после внедрения разрабатываемого программного обеспечения. За счет упрощения коммуникации между преподавателем и студентом, повышением общей мобильности системы, мы получим полностью удовлетворяющий нашим требованиям готовый продукт.

Таким образом, методологией IDEF0 представлена будущая структура системы, с обозначенными требованиями и необходимым функционалом новой системы, разбитой на составляющие подфункции.

## **Описание организации структур, хранящих данные.**

В ходе проектирования приложения в качестве способа организации входных данных были выбраны классы. Выбор обусловлен лёгкой программной функциональностью и расширенными возможностями. Для защиты персональных данных клиентов использовано хеширование паролей с «солью», что соответствует современным стандартам безопасности [8]. Описание используемых структур представлено ниже.

Таблица 2.1 – Описание полей класса Date

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Уровень доступа | Тип данных | Назначение поля |
| day | private | int | День |
| month | int | Месяц |
| year | int | Год |

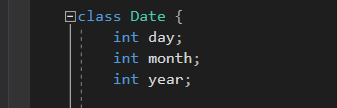


Рисунок 2.6 – Класс Date

Таблица 2.2 – Описание класса User

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Уровень доступа | Тип данных | Назначение поля |
| login | private | string | Логин |
| salted\_hash\_password | string | Хешированный пароль |
| salt | string | «Соль» |
| isAdmin | bool | Проверка является ли администратором |
| access | bool | Проверка имеет ли пользователь доступ |
| Id | int | Порядковый номер записи в файле |

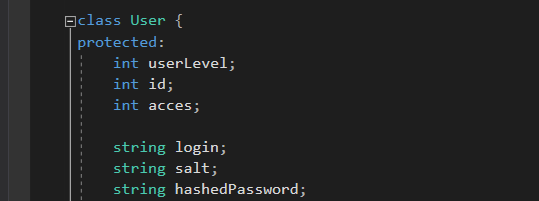


Рисунок 2.7 – Класс User

Таблица 2.3 – Описание класса Student

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Уровень доступа | Тип данных | Назначение поля |
| name | private | string | Имя студента |
| secondname | string | Фамиля студента |
| surname | string | Отчество студента |
| group | int | Группа студента |
| course | int | Курс студента |

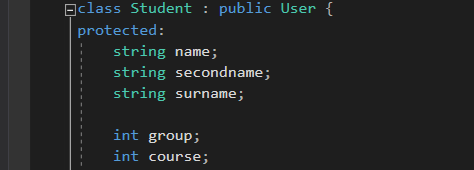


Рисунок 2.8 Класс Student

Таблица 2.4 – Описание класса StudentCourseWork

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Уровень доступа | Тип данных | Назначение поля |
| deadLinePointsMarks | private | int[const int] | Оценки по к.т. |
| courseWorkTheme | string | Тема работы |
| courseWorkStorageLink | string | Ссылка на ресурс с работой |

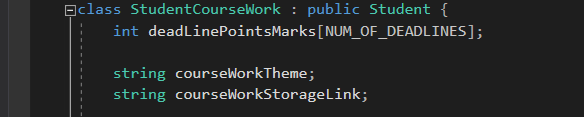


Рисунок 2.9 Класс StudentCourseWork

Для объединения данных используются массивы std::vector.

Для хранения и манипулирования данными в программе используются текстовые файлы с расширением «.txt».

Логины, пароли и сведения о статусе учетных записей хранятся в файле «REG\_STUDENTS.txt». Сведения о клиентах располагаются в файле «DATA\_STUDENTS.txt».

## **Разработка перечня пользовательских функций программы.**

Ниже приведён перечь функций и методов, разработанных для корректной работы программного средства организации выполнения курсовой работы студентами группы.

void FilePreparation(); //Загрузить необходимые для старта файлы

void InitializeMenuLoginEntering(); //Меню выбора роли для входа

void CheckAdmins(); //Проверить, сколько админов в системе

bool UserAuthorizationMenu(bool); //Меню ввода пароля/логина

bool CheckRegistration(bool isAdmin, string& login, string& password); //Проверить, существует ли побдобная запись

extern void AdminFunctionsMenu(); //Вывести функционал администратора

void RefreshMenu(const string, int); //Обновить заголовок

void AccountsListOperations(); //Предоставить возможные операции с учетными записями

void PrintAccountssFromFile(); //Вывести информацию об учетных записях

void AddAccountMenu(); //Меню добавления учетной записи

void EditAccountsFromArrayMenu(); //Меню изменения учетной записи

void DeleteAccountsFromArrayMenu(); //Меню удаления учетной записи

void GiveAccesStudents(); //Меню предоставления доступа для входа

void ShowStudentsDataTable(); //Показать таблицу студентов

void SetCourseDeadlines(); //Меню установки дедлайнов

void SetStudentsMarks(); //Меню рецензирования студента

void SetStudentCourseTheme(); //Меню установки темы курсовой

void IndividualTask(); //Индивидуальное задание

extern void UserFunctionsMenu(); //Меню пользователя

void PrintAccountData(); //Вывести данные текущей учетной записи

void EditAccountData(); //Изменить данные текущей учетной записи

void SetCourseWorkLink(); //Меню установки ссылки на курсовую

void GetStudentByParam(); //Поиск студента по параметрам

string Student::getName() const { return name; } //получить имя

string Student::getSecondName() const { return secondname; } //получить фамилию

string Student::getSurname() const { return surname; } //получить отчество

int Student::getGroup() const { return group; } //получить группу

int Student::getCourse() const { return course; } //получить курс

string User::getSalt() const { return salt; } //получить соль

string User::getHashedPassword() const { return hashedPassword; } //получить хэшированный пароль

int User::getAcces() const { return acces; } //получить доступ

void User::setAcces(int newAcces) { acces = newAcces; } //установить доступ

string StudentCourseWork::getCourseWorkTheme(); //получить тему

string StudentCourseWork::getCourseWorkLink(); //получить ссылку

int StudentCourseWork::getMark(int index); //получить оценку

void StudentCourseWork::setMark(int deadLineIndex, int mark); //установить оценку

void StudentCourseWork::setCourseWorkTheme(); //Установить тему

## **Разработка схем алгоритмов работы программы**

Алгоритм – набор последовательных инструкций, которые описывают порядок поведения программы для достижения нужной цели [4].

На рисунке 2.10 представлен алгоритм функции IndividualTask, которая вызывается администратором и выводит информацию о неуспевающих студентах.

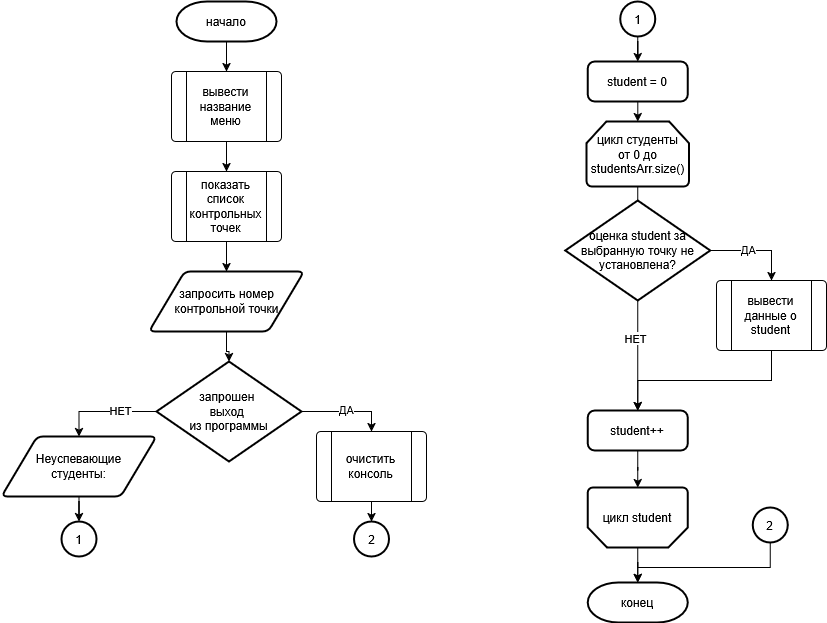


Рисунок 2.11 – Схема алгоритма функции IndividualTask

На рисунке 2.12 представлен алгоритм функции GiveAccesStudents. Данная функция может вызываться администратором для ответа на поданые заяки на регистрацию.

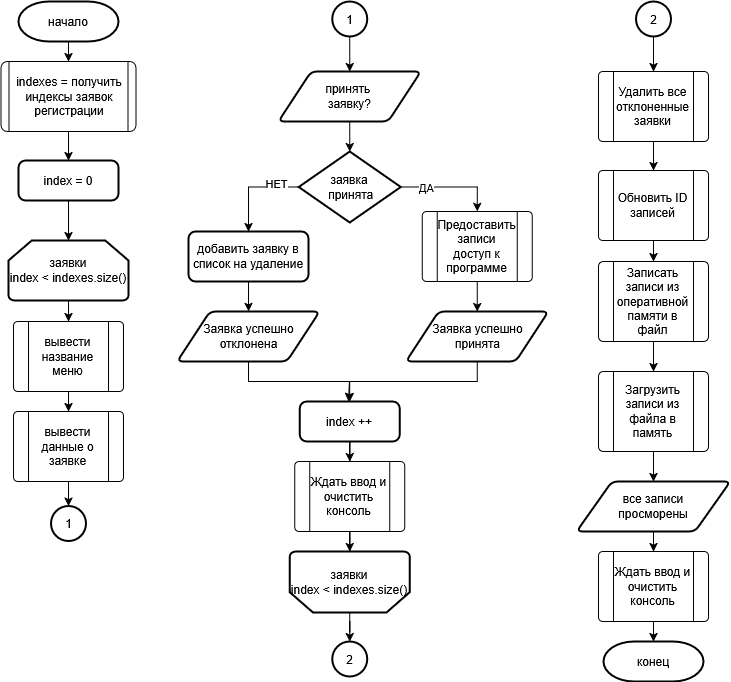


Рисунок 2.12 – Схема алгоритма функции GiveAccesStudents

На рисунке 2.13 представлена блок-схема основной бизнес логики программы.

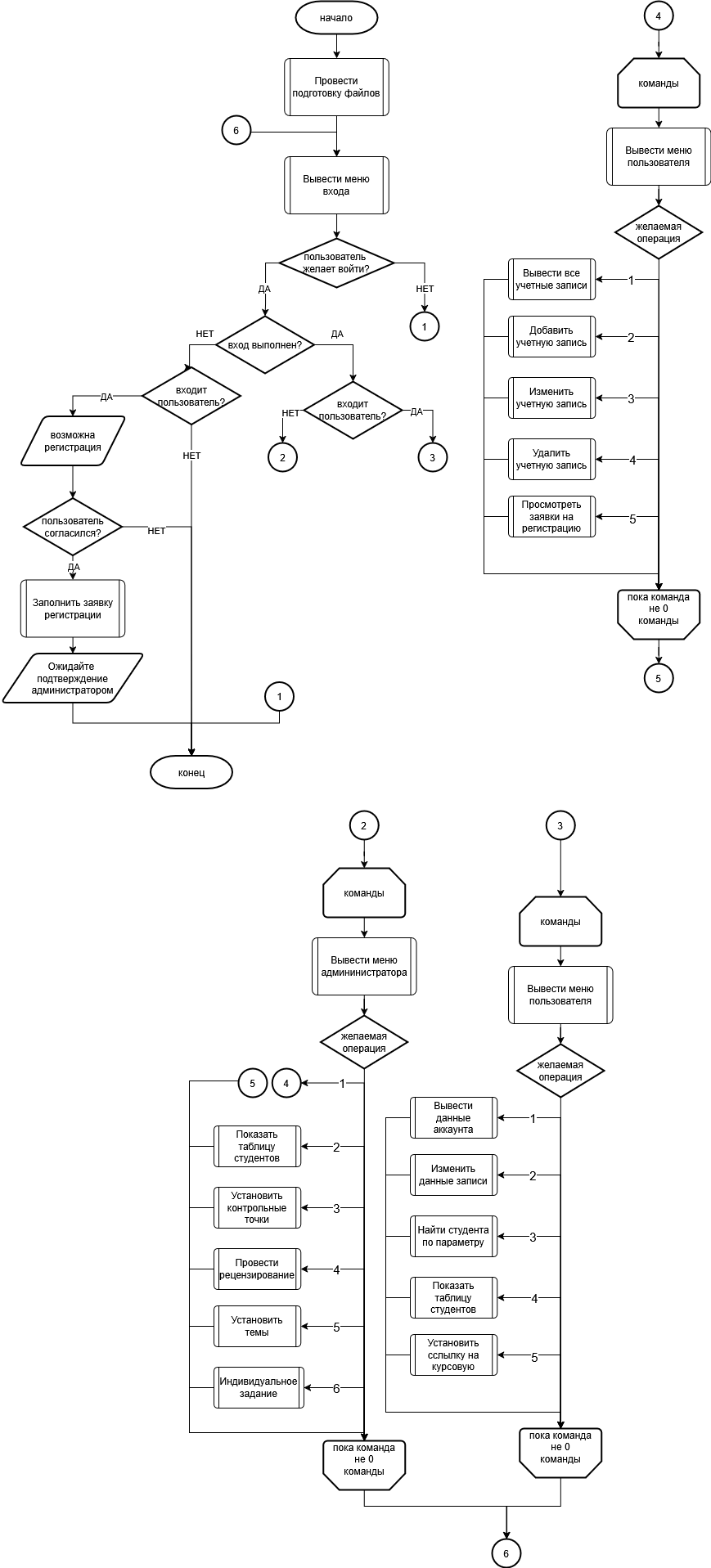


Рисунок 2.13 Схема бизнес логики программы

3 **ТЕСТИРОВАНИЕ И ПРОВЕРКА РАБОТОСПОСОБНОСТИ ПРОГРАММНОГО СРЕДСТВА**

Тестирование программного обеспечения (ИС) для управления работой студентов - это процесс проверки соответствия системы установленным требованиям, а также выявление и устранение возможных ошибок.

По результатам тестирования было подтверждено, что информационная система управления работой студентов полностью соответствует заявленным функциональным требованиям. Программа предусматривает решение следующих исключительных ситуаций:

* при регистрации ввод логина, который уже зарегистрирован;
* некорректный ввод пароля;
* некорректный ввод строковых данных;
* некорректный ввод числовых значений.
* некорректный ввод личных данных

При регистрации нового пользователя происходит проверка на наличие логина (рисунок 3.1). При вводе пароля запрашивается не менее 5 символов, в противном случае выводится соответствующее сообщение (рисунок 3.2).

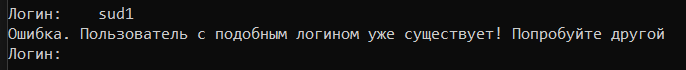


Рисунок 3.1 – Пример ввода существующешго логина

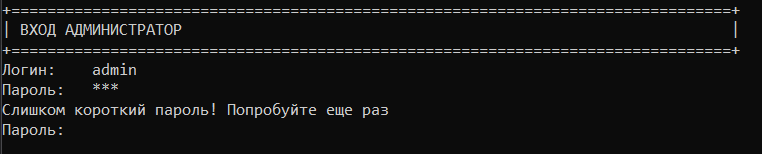


Рисунок 3.2 – Сообщение об ошибке ввода пароля

Если во время авторизации была выбрана неверная роль, введены неверный логин или пароль программа выдаст сообщение (рисунок 3.3).

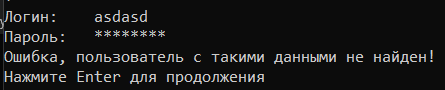


Рисунок 3.3 – Сообщение об ошибке авторизации

Рассмотрим исключительные ситуации, возникающие при работе пользователя с системой. При попытке указать неверные личные данные (нелогичные символы или данные) пользователю выведется следующая ошибка (рисунок 3.4). При вводе ссылки на ресурс курсовой работы система проверяет, соответствует ли она базовому формату ссылок (рисунок 3.5)



Рисунок 3.4 – Сообщение ошибки заполнения личных данных



Рисунок 3.5 – Сообщение ошибки заполнения личных данных

Рассмотрим исключительные ситуации, которые могут возникать при работе администратора с системой. При попытке просмотра запросов на регистрацию, которые отсутствуют выводится соответствующее сообщение. На рисунке 3.5 приведен пример такой ситуации при отсутствии заявок. При редактировании и удалении данных запрашивается один из параметров записи в списке, если таковой не существует выдается соответствующее сообщение. (рисунок 3.6). Если администратор решит удалить свою собственную запись, программа выведет соответствующую ошибку (рисунок 3.7). При заполнении контрольных точек, программа проверяет хронологический порядок точек, а так же их существование (рисунок 3.8). Система проверяет, корректна ли вводимая оценка за контрольную точку (рисунок 3.9). Преподаватель не может установить оценку или тему курсовой работы. Пример обработки такой ситуации представлен на рисунке 3.10.

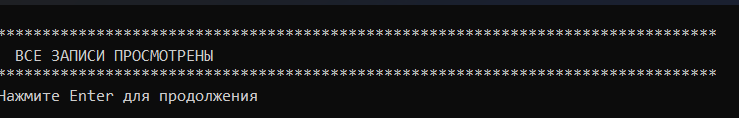


Рисунок 3.5 – Пример просмотра заявок регистрации при их отсутствии

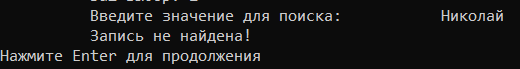


Рисунок 3.6 – Пример редактирования несуществующих данных



Рисунок 3.7 – Пример попытки само удалить запись (по индексу)

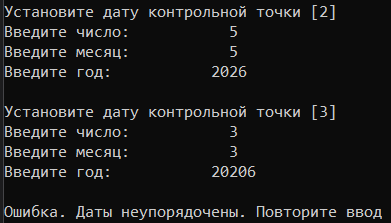


Рисунок 3.8 – Пример ввода неправильно упорядоченных дат

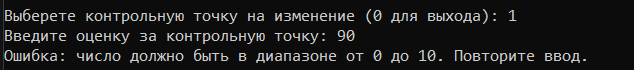


Рисунок 3.9 – Пример неправильного заполнения результатов точки



Рисунок 3.10 – Пример попытки администратора установить свою тему

Проведение тестирования программного обеспечения подчеркивает его важность и преимущества, способствует надежности и безопасности программного средства и повышает удовлетворенность пользователей.

# **4 ИНСТРУКЦИЯ ПО РАЗВЁРТЫВАНИЮ ПРИЛОЖЕНИЯ И СКВОЗНОЙ ПРИМЕР**

## **4.1 Главное меню**

При запуске программы открывается главное меню, в котором пользователь может сделать выбор: войти в систему как пользователь или администратор или выйти из программы. На рисунке 4.1.1 показано главное меню.

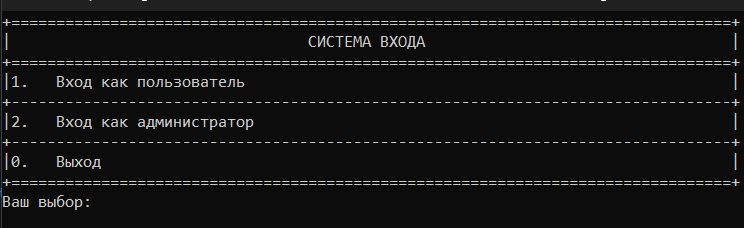


Рисунок 4.1.1 – Главное меню

При выборе авторизации перед пользователем появляется меню (меню варьируется в зависимости от выбора авторизации), показанное на рисунке 4.1.2 Из этого меню можно авторизоваться в качестве пользователя и администратора, для чего необходимо ввести данные ранее зарегистрированной учетной записи. Если данные введены правильно, откроется модуль пользователя или модуль администратора, в зависимости от выбранной на данном этапе роли.

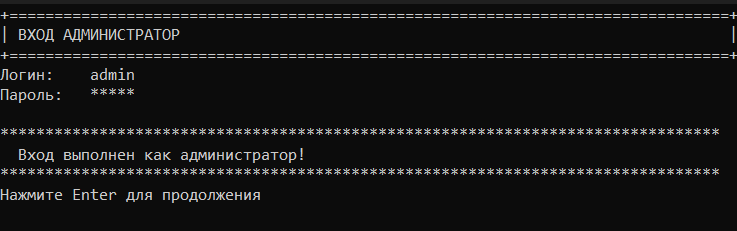


Рисунок 4.1.2 – Меню авторизации

В программе реализован альтернативный вариант регистрации. Если на момент регистрации логина пользователя не существовало в программе, будет предложено создать новую учетную запись, которую позже должен одобрить администратор (рисунок 4.1.3)

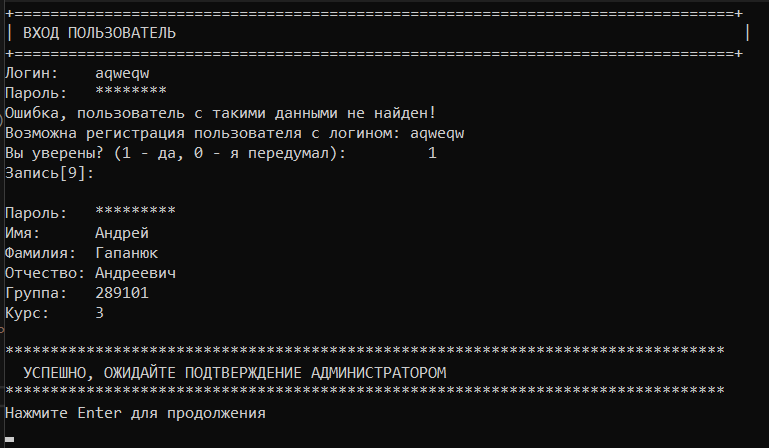


Рисунок 4.1.2 – Альтернативное меню регистрации

Главное меню является ключевым и незаменимым элементом каждой программы. Это меню содержит фундаментальную информацию, с которой необходимо ознакомиться до начала использования программного продукта.

## **4.2 Модуль администратора**

После успешной аутентификации администратору открывается меню администратора, которое включает в себя следующие пункты: работа с учетными записями, просмотр таблицы данных, установка контрольных точек, проведение рецензирования работ, установка тем курсовых работ, выполнение индивидуального задания и выход из учетной записи администратора. Как показано на рисунке 4.2.1, отображается меню администратора.

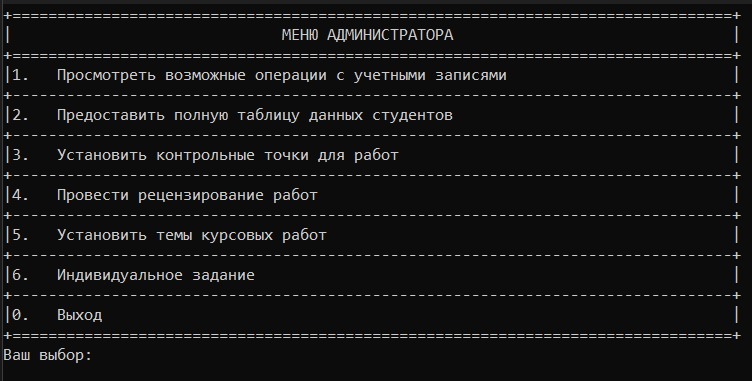


Рисунок 4.2.1 – Меню администратора

При выборе работы с учетными записями пользователей (рисунок 4.2.2), предоставления таблицы студентов (рисунок 4.2.3), работы с установкой контрольных точек (рисунок 4.2.4), пункта рецензирования курсовых работ (рисунок 4.2.5) или пункта установки тем курсовых работ (рисунок 4.2.6) и выполнения индивидуального задания (рисунок 4.2.7) открываются соответствующие меню.

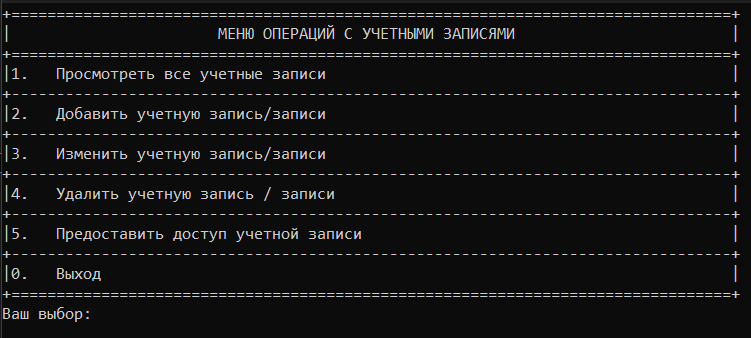


Рисунок 4.2.2 – Меню операций с учетными записями пользователей

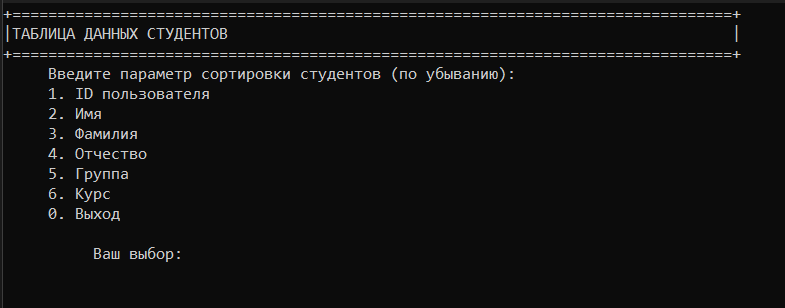


Рисунок 4.2.3 – Таблица данных студентов

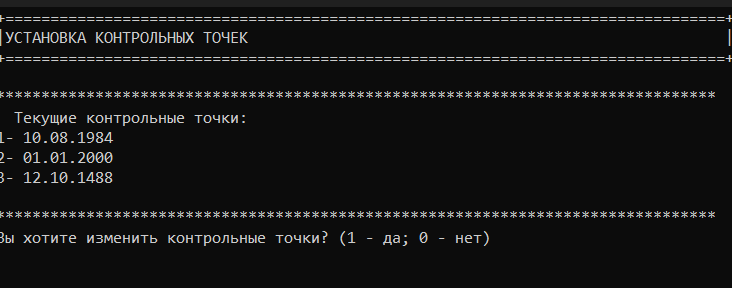


Рисунок 4.2.4 – Установка контрольных точек

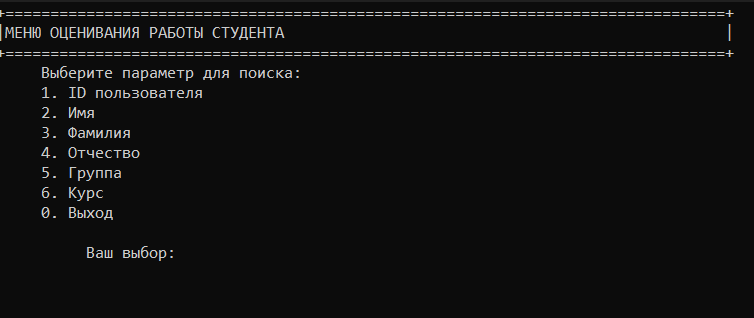


Рисунок 4.2.5 – Меню оценивания работы студента

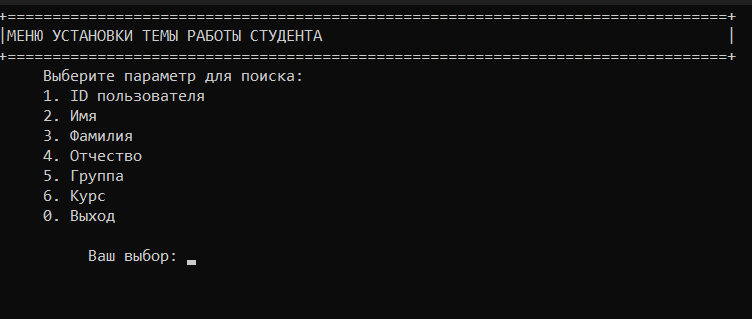


Рисунок 4.2.6 – Меню выбора темы курсовой работы студенту

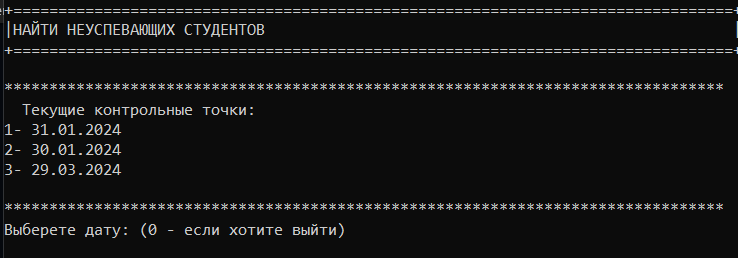


Рисунок 4.2.7 – Меню выбора темы курсовой работы студенту

Пример просмотра всех учетных записей приведен на рисунке 4.2.8.

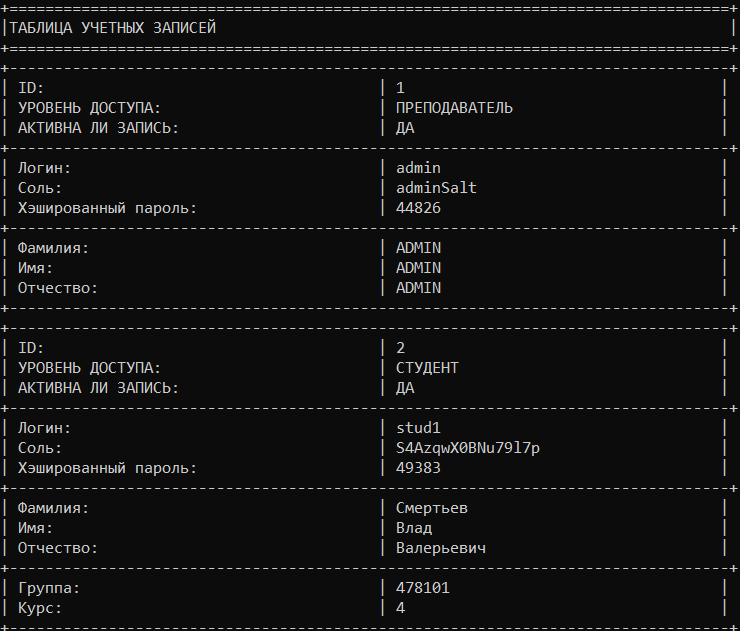


Рисунок 4.2.8 – Меню просмотра всех учетных записей

После выбора действия «Добавить новую учетную запись» администратор получает возможность зарегистрировать в системе нового пользователя или администратора (см. Рисунок 4.2.9). Для выполнения этого процесса необходимо указать уникальный логин, правильный пароль и назначенную роль.

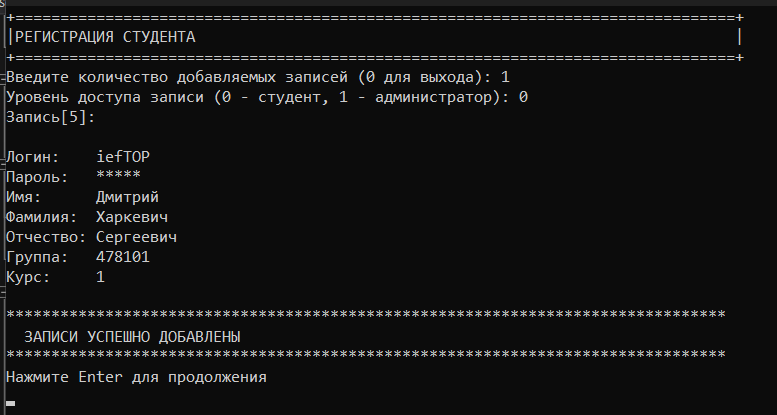


Рисунок 4.2.9 – Пример добавления учетной записи преподавателя

При выборе действия по редактированию данных учетной записи запрашивается и отображается на экране номер записи, данные которой необходимо изменить. После этого администратор может заполнить данные записи. Как показано на рисунке 4.2.10а и 4.2.10б, пример редактирования данных представлен для учетных записей.

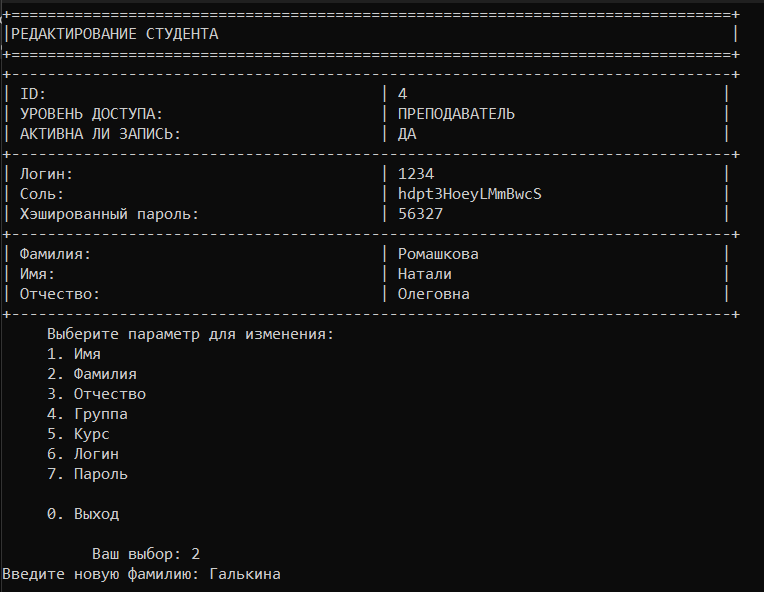


Рисунок 4.2.10а – Пример редактирования данных учетной записи (выбор поля для редактирования)

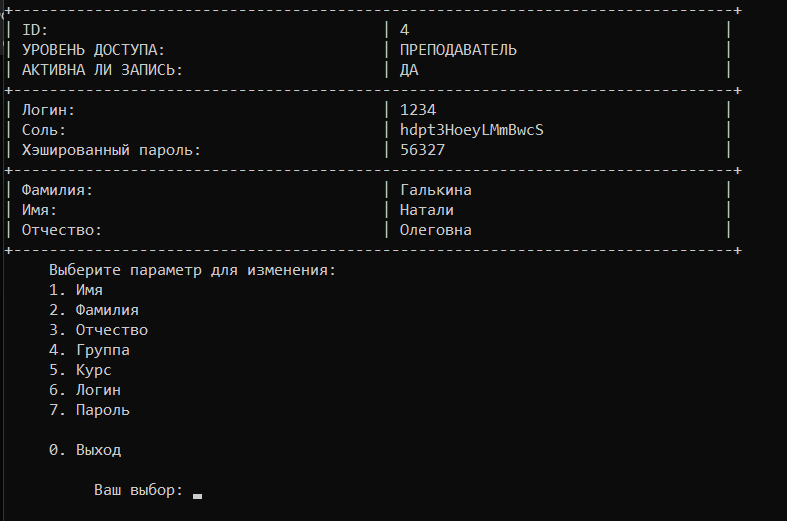


Рисунок 4.2.10б – Пример редактирования данных учетной записи (после успешного редактирования)

После выбора опции «Удалить учетную запись» администратору будет предложено выбрать запись, которая затем отобразится на экране. Удаление учетной записи произойдёт только с получения согласия пользователя. Пример удаления учетной записи показан на рисунке 4.2.11.

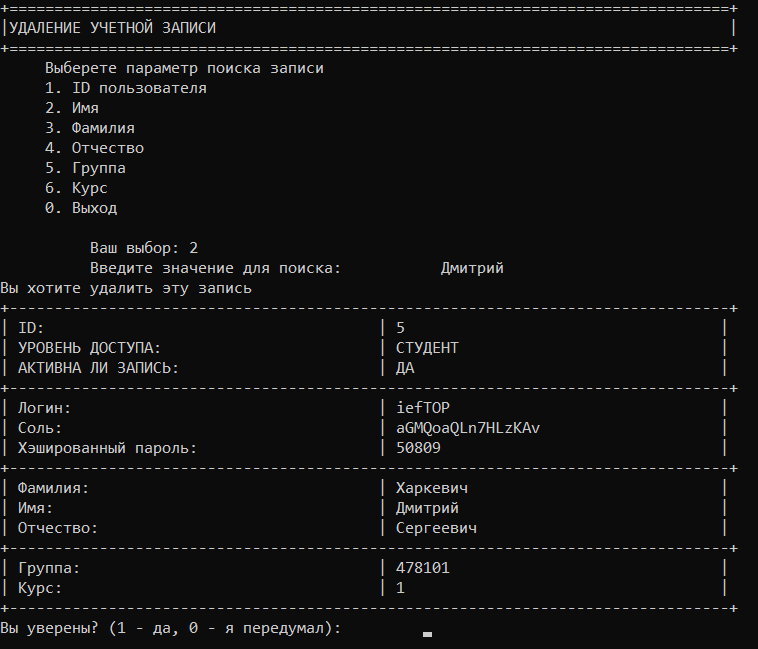


Рисунок 4.2.11 – Пример удаления учетной записи

В пункте «Предоставить доступ учетной записи» администратору предоставляется список студентов, чья регистрация еще не была подтверждена. На рисунке 4.2.11 представлен процесс просмотра такой заявки.

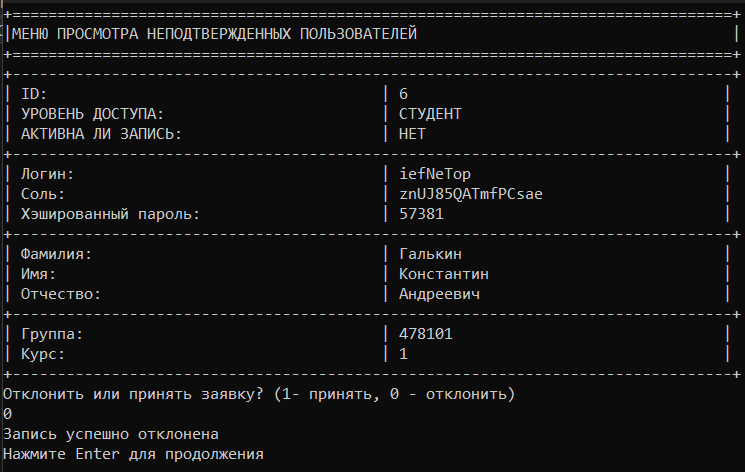


Рисунок 4.2.11 – Пример обработки заявки от студента

В меню работ с учетными записями и данными студентов, администратору доступны функции поиска, в меню предоставления данных студентов администратор может отсортировать значения на выбор.

Данные клиентов могут быть найдены и отсортированы по фамилии, по имени, по отчеству, по номеру в файле, по группе, по курсу. На рисунке 4.2.12а и 4.2.12б представлен пример поиска клиента по имени. На рисунке 4.2.13 – результат сортировки студентов по имени.

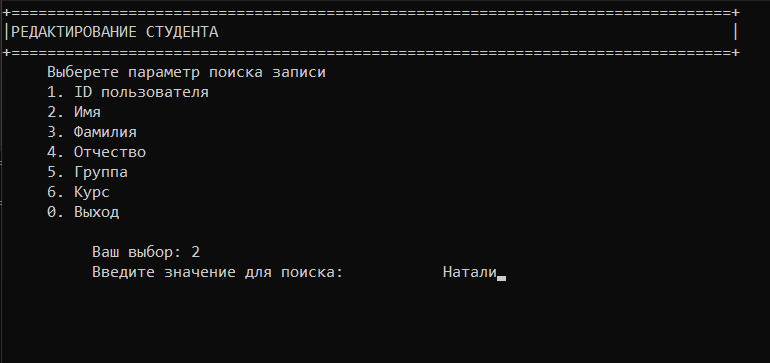


Рисунок 4.2.12а – Пример поиска учетной записи по имени (предварительный этап)

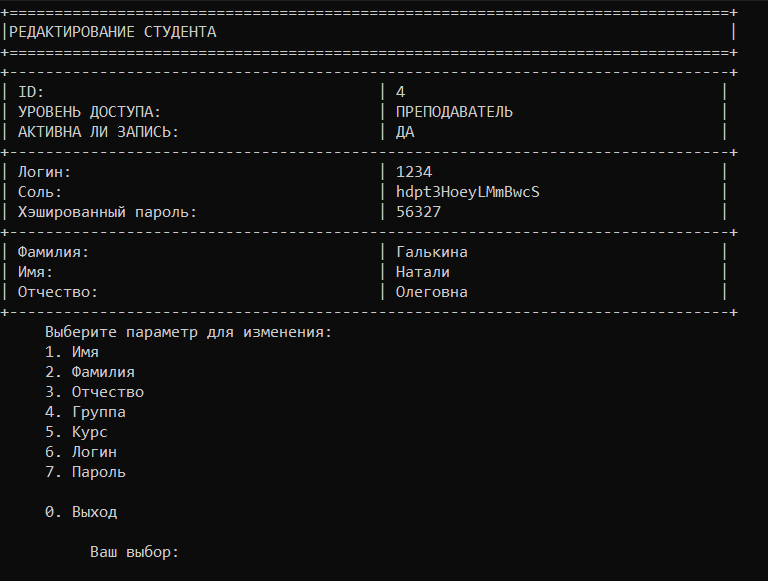


Рисунок 4.2.12б – Пример поиска учетной записи по имени (вывод поиска по имени)

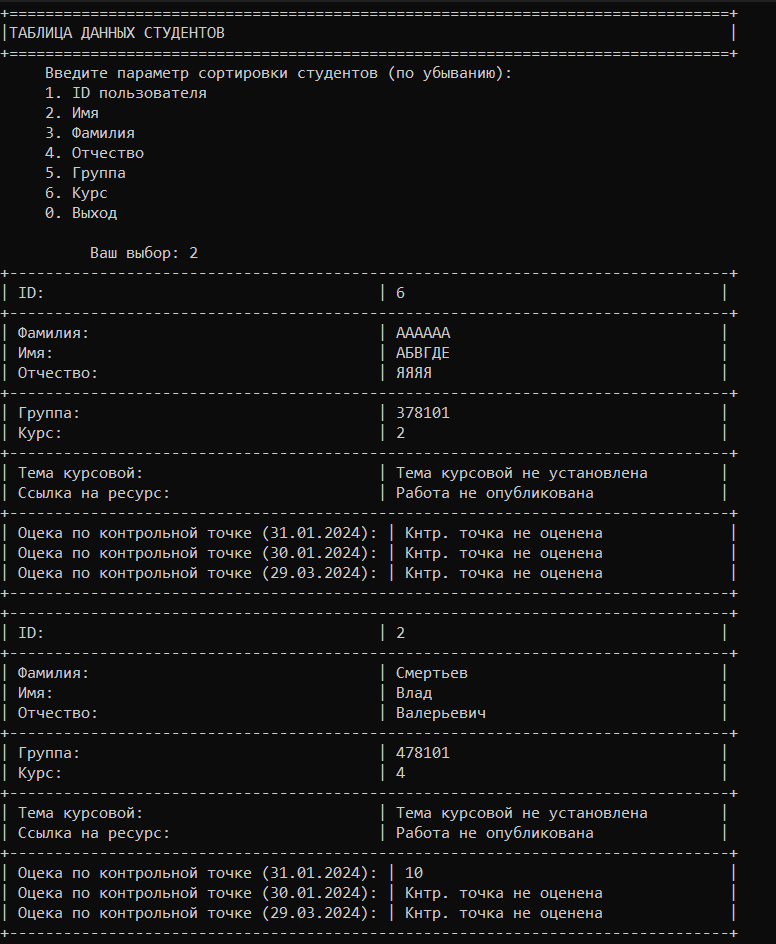


Рисунок 4.2.13 – Пример сортировки студентов по имени

Преподаватель имеет доступ к изменению контрольных точек с указанием и просмотром даты по каждой из них. Пример редактирования контрольных точек представлен на рисунке 4.2.14

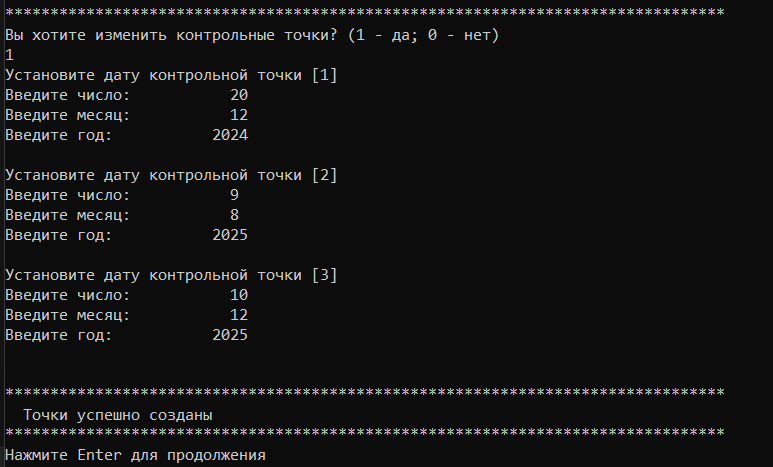


Рисунок 4.2.14 – Пример установки новых контрольных точек

Через меню рецензирования преподаватель может с помощью поиска получить студента и провести его оценивание по каждой из контрольных точек. Пример рецензирования контрольных точек представлен на рисунке 4.2.14а и 4.2.14. Аналогично для установки тем курсовых работ (рисунок 4.2.15а и 4.2.15б)

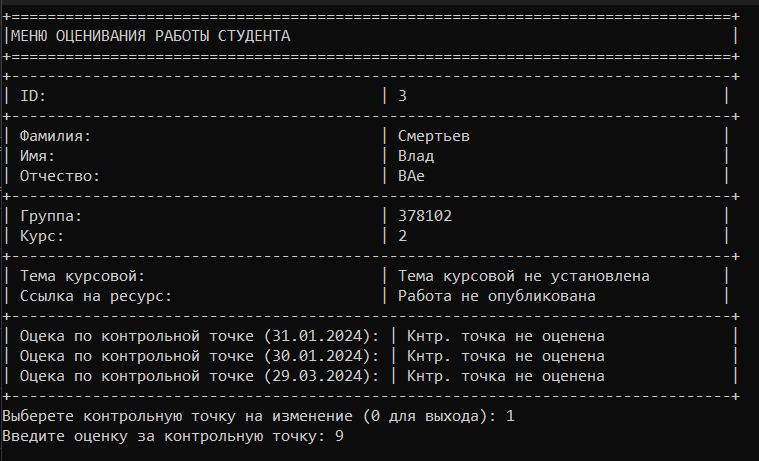


Рисунок 4.2.14а – Пример рецензирования контрольной точки (состояние до)

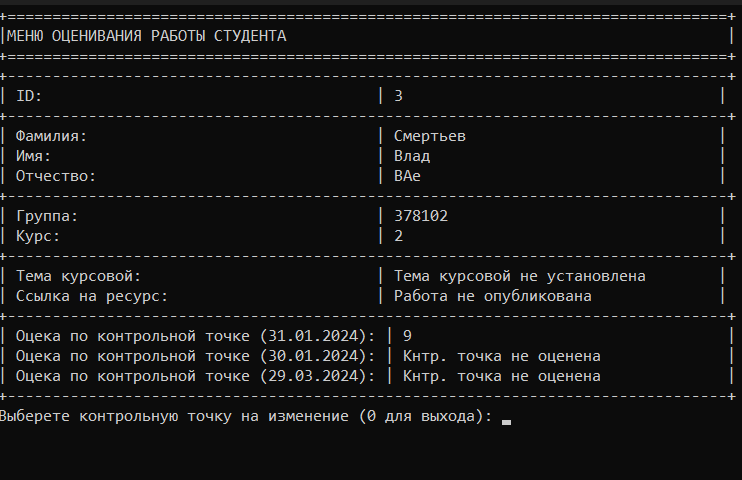


Рисунок 4.2.14б – Пример рецензирования контрольной точки (обновленное состояние студента)

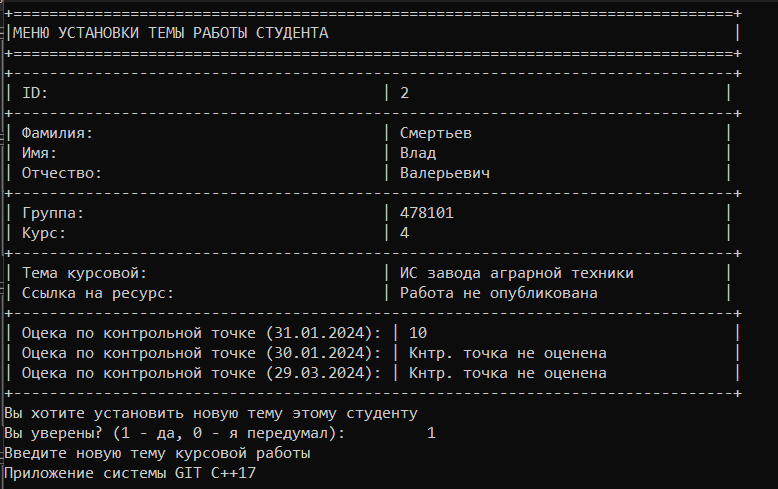


Рисунок 4.2.15а – Пример обновления темы студента (состояние до)

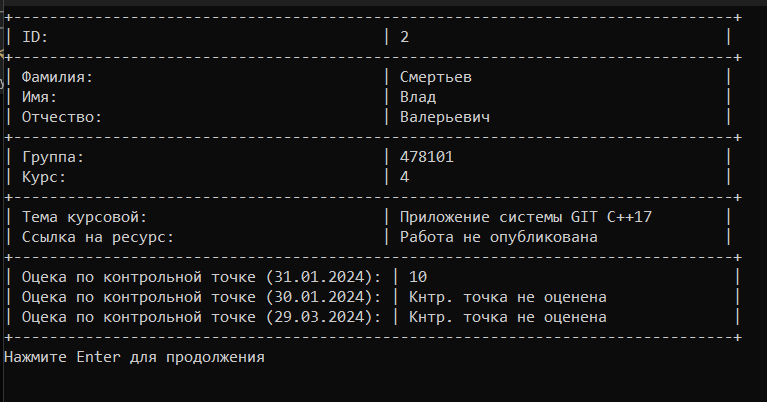


Рисунок 4.2.15б – Пример обновления темы студента (состояние после)

В качестве индивидуального задания было необходимо реализовать меню отбора неуспевающих студентов по определенной дате. Реализация функции продемонстрирована на рисунке 4.2.16.

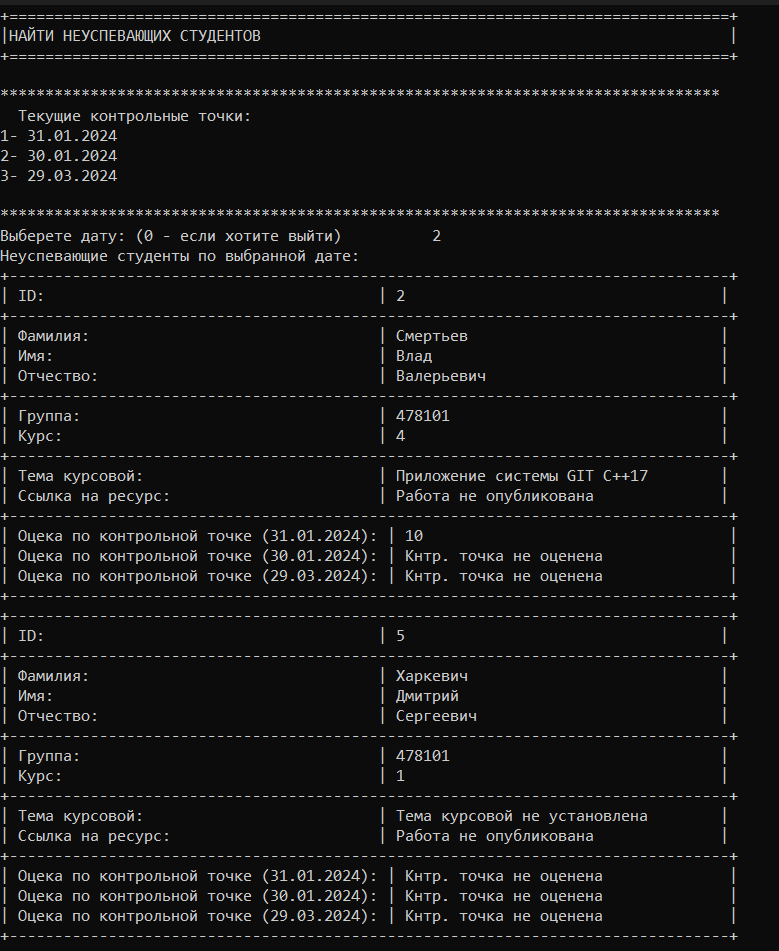


Рисунок 4.2.16 - Реализация индивидуального задания

Очевидно, что функция возврата в меню администратора присутствует в каждом меню второго уровня в модуле администратора. Процесс выхода из аккаунта и доступа к главному меню осуществляется из административного меню.

Модуль администратора включает в себя множество функций, облегчающих работу со счетами, данными студентов, их курсовыми работами.

В разработанной программе пользователю и администратору представлен широкий функционал, который позволит им решить поставленные задачи. Очевидно, что использование ясного и понятного языка не вызовет сложностей. Использование данной программы позволит повысить эффективность работы в высших учреждениях образования.

## **4.3 Модуль пользователя**

При входе в аккаунт пользователя отображается меню пользователя, которое содержит следующие пункты: просмотреть личные данные, заполнить личные данные, подать заявку на изготовление карты, просмотреть оставленные заявки, выйти из аккаунта пользователя. На рисунке 4.3.1 представлено меню пользователя.

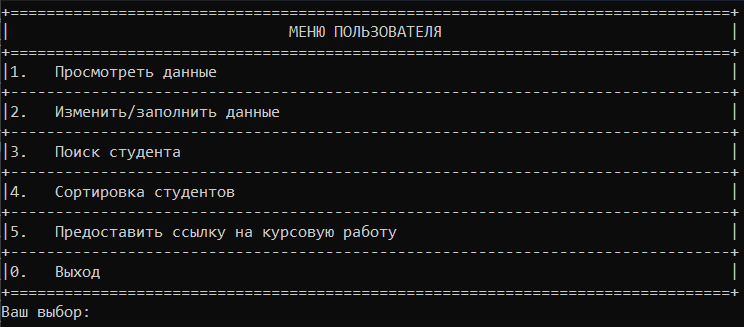


Рисунок 4.3.1 – Меню пользователя

При выборе действия просмотра личных данных и наличии самых данных они выводятся на экран в виде строки таблицы (рисунок 4.3.2). Поле “Изменить/заполнить данные” повторяет структуру изменения полей записи для администратора, описанной выше, однако выбор записи на изменение не предоставляется.

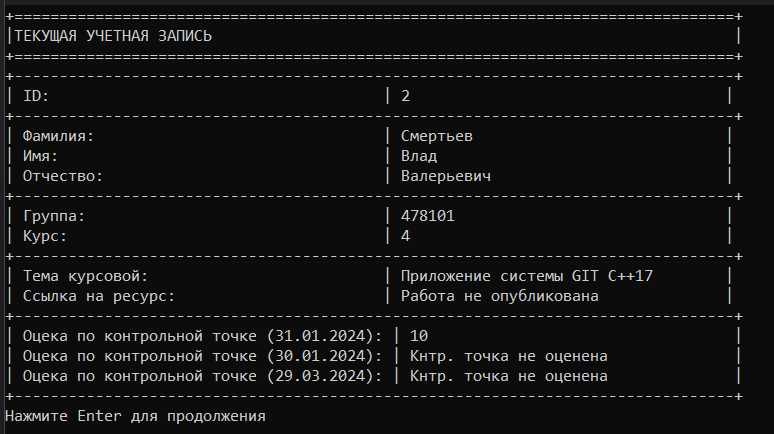


Рисунок 4.3.2 – Вывод личных данных пользователя

Пример меню поиска студента представлен на рисунке 4.3.3. Пример сортировки списка студентов показан на примере 4.3.4. Пользователь так же может предоставить (изменить) ссылку на свою работу (рисунок 4.3.5).

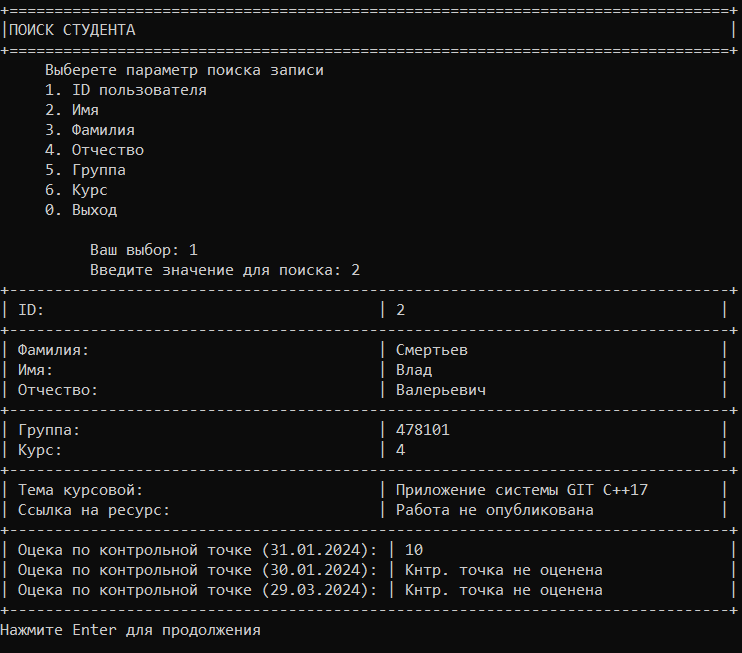


Рисунок 4.3.3 – Пример меню поиска студента

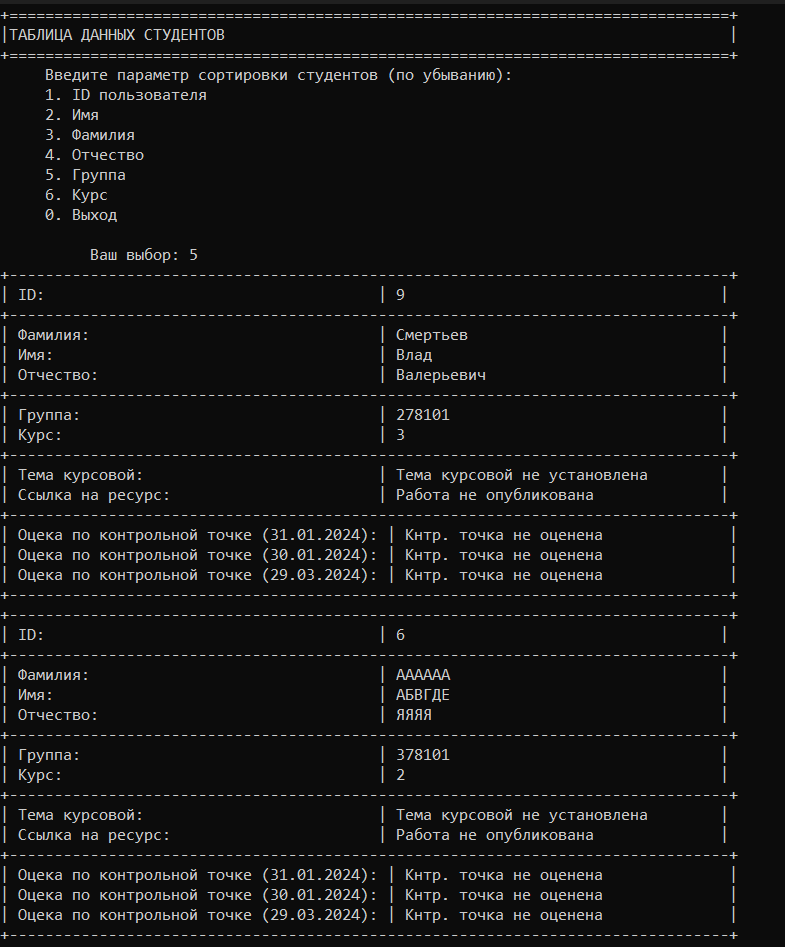


Рисунок 4.3.4 – Пример сортировки списка студентов

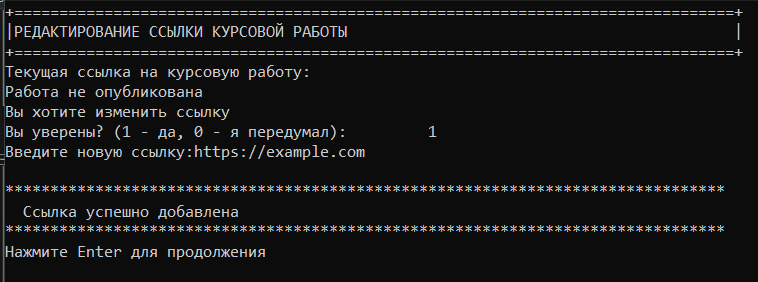


Рисунок 4.3.5 – Изменение ссылки на ресурс с курсовой

После выбора «Выйти из аккаунта пользователя» пользователь вернется в главное меню.

Модуль пользователя поддерживает все необходимые функции, которые позволяют ему быстро и эффективно использовать программу. В пользовательском меню формулировки возможных действий кратко и четко изложены, что снижает риск недопонимай пользователя.

# **Заключение**

В ходе разработки курсового проекта было создано программное средство, предназначенное для организации и контроля выполнения курсовых работ студентами группы. Были изучены требования к системе, определены ключевые функции, а также проанализированы особенности учебного процесса, что позволило сформировать четкий технический план проекта.

В процессе разработки детально исследованы организационные процессы, связанные с выполнением курсовых работ, включая распределение тем, контроль сроков, взаимодействие с преподавателями и оценку результатов. Это обеспечило создание удобного интерфейса для управления информацией о студентах, темах работ, этапах выполнения, сроках сдачи и оценках. Реализованы функции добавления, редактирования и удаления данных, а также поиска и сортировки информации по различным критериям, таким как ФИО студента, тема работы, статус выполнения и оценка. Для удобства пользователей предусмотрены уведомления о приближающихся дедлайнах и возможность формирования отчетов.

Программное средство разработано на языке C++ с учетом удобства взаимодействия пользователя с системой. Для предотвращения ошибок ввода реализованы проверки данных, а в случае некорректных действий пользователь получает сообщение с пояснением и возможностью повторного ввода. Логика работы программы отражена в функциональных алгоритмах, представленных в пояснительной записке. Для наглядности архитектуры системы разработана модель бизнес-процессов в нотации IDEF0, демонстрирующая взаимодействие между студентами, преподавателями и системой.

Данное программное средство позволяет автоматизировать процесс распределения и контроля курсовых работ, упростить взаимодействие между студентами и преподавателями, обеспечить прозрачность и своевременность выполнения этапов работы, а также формировать аналитические отчеты для оценки прогресса группы. Система удобна в использовании как для студентов, так и для преподавателей, что делает её эффективным инструментом организации учебного процесса. В перспективе возможно расширение функционала, включая интеграцию с электронными образовательными платформами и добавление модуля онлайн-консультаций.

Таким образом, поставленные цели и задачи выполнены: разработано надежное и удобное программное средство, способствующее эффективной организации выполнения курсовых работ в учебной группе.

# **Список использованных источников**

[1] МакГоуэн К. Методология структурного анализа и проектирования SADT. - М., 1993. - 240 с.

[2] Страуструп, Б. Язык программирования C++. Специальное издание. / пер. с англ. – М. : Вильямс, 2012. – 1136 с.

[3] Microsoft Docs. Документация по Visual Studio 2022 [Электронный ресурс]. - Режим доступа: <https://docs.microsoft.com/ru-ru/visualstudio/>

[4] A Web-based Application for the Management of Seminar Assignments // arXiv.org, 2017. - Режим доступа: <https://arxiv.org/ftp/arxiv/papers/1706/1706.09266.pdf>

[5] Automata Tutor v3 // arXiv.org, 2020. - Режим доступа: <http://arxiv.org/pdf/2005.01419.pdf>

[6] A Web-based Application for the Management of Seminar Assignments. - 2017. - Режим доступа: <https://arxiv.org/ftp/arxiv/papers/1706/1706.09266.pdf>

[7] Improving Feedback from Automated Reviews of Student Spreadsheets. 2023. - Режим доступа: <https://arxiv.org/ftp/arxiv/papers/2311/2311.10728.pdf>

[8] Gradeer: An Open-Source Modular Hybrid Grader. - 2021. - Режим доступа: <https://arxiv.org/pdf/2102.09400.pdf>

# **ПРИЛОЖЕНИЕ А (обязательное)**

## **Отчет о проверке на заимствование в системе «Антиплагиат»**

# 

# Рисунок А.1 –Отчет о проверке на заимствование **ПРИЛОЖЕНИЕ Б (обязательное)**

## **Листинг кода алгоритмов, реализующих основную бизнес-логику**

void PrintAccountssFromFile() {

const string HEADER = "ТАБЛИЦА УЧЕТНЫХ ЗАПИСЕЙ";

HeaderSecondLevel(HEADER);

for (int i = 0; i < studentsArray.size(); i++) {

AccoutTable(studentsArray[i]);

}

WaitEnterInput();

}

void AddAccountMenu() {

const string HEADER = "РЕГИСТРАЦИЯ СТУДЕНТА";

HeaderSecondLevel(HEADER);

RegistrateStudentInFile();

string message = "ЗАПИСИ УСПЕШНО ДОБАВЛЕНЫ";

LogMessage(message);

WaitEnterInput();

}

void EditAccountsFromArrayMenu() {

const string HEADER = "РЕДАКТИРОВАНИЕ СТУДЕНТА";

HeaderSecondLevel(HEADER);

vector<int> indexes;

do {

indexes = FindStudentByParam();

if (indexes.size() == 1 && indexes[0] == -1)

break; //если был выбран выход

for (int i = 0; i < indexes.size(); i++) {

int index = indexes[i];

RefreshMenu(HEADER,index);

studentsArray[index].StudentEdit();

}

ClearTerminal();

HeaderSecondLevel(HEADER);

}while (true);

StudentFileRewrite();

}

void DeleteAccountsFromArrayMenu() {

const string HEADER = "УДАЛЕНИЕ УЧЕТНОЙ ЗАПИСИ";

HeaderSecondLevel(HEADER);

vector<int> indexes;

int delChoice;

do {

indexes = FindStudentByParam();

if (indexes.size() == 1 && indexes[0] == -1)

break; //если был выбран выход

for (int i = 0; i < indexes.size(); i++) {

int index = indexes[i];

if (studentsArray[index].getLogin() == userAccountLink->getLogin()) {

cout << "Ошибка, самоудаление невозможно!\n";

WaitEnterInput();

break;

}

cout << "Вы хотите удалить эту запись\n";

AccoutTable(studentsArray[index]);

if (GetUserApprove()) {

DeleteStudentArray(index);

RefreshStudentsId();

cout << "Запись успешно удалена" << endl;

WaitEnterInput();

}

}

ClearTerminal();

HeaderSecondLevel(HEADER);

} while (true);

StudentFileRewrite();

}

void ShowStudentsDataTable() {

const string HEADER = "ТАБЛИЦА ДАННЫХ СТУДЕНТОВ";

HeaderSecondLevel(HEADER);

vector<int> indexes = SortIndexes();

for (int i = 0; i < indexes.size(); i++) {

int index = indexes[i];

StudentWorkCourseTable(studentsArray[index]);

}

if (indexes.size() != 0)

WaitEnterInput();

else

ClearTerminal();

}

void SetCourseDeadlines() {

const string HEADER = "УСТАНОВКА КОНТРОЛЬНЫХ ТОЧЕК";

HeaderSecondLevel(HEADER);

ShowDeadLinesList();

cout << "Вы хотите изменить контрольные точки\n";

if (GetUserApprove()) {

do {

for (int i = 0; i < NUM\_OF\_DEADLINES; i++) {

cout << "Установите дату контрольной точки [" << i + 1 << "]" << endl;

courseDeadLinePoints[i].SetDate();

cout << endl;

}

} while (!CheckDates());

string message = "Точки успешно созданы";

LogMessage(message);

SaveDeadLinesInFile();

WaitEnterInput();

}

else

ClearTerminal();

return;

}

bool SelectMark(int index) {

int mark;

int deadLineIndex;

cout << "Выберете контрольную точку на изменение (0 для выхода): ";

deadLineIndex = GetIntegerInput(0,NUM\_OF\_DEADLINES);

if (deadLineIndex == 0) return false;

deadLineIndex--;

cout << "Введите оценку за контрольную точку: ";

mark = GetIntegerInput(0, 10);

studentsArray[index].setMark(deadLineIndex, mark);

return true;

}

//Установаить оценки студентам по контрольным точкам

void SetStudentsMarks() {

const string HEADER = "МЕНЮ ОЦЕНИВАНИЯ РАБОТЫ СТУДЕНТА";

HeaderSecondLevel(HEADER);

vector<int> indexes;

do {

ClearTerminal();

HeaderSecondLevel(HEADER);

indexes = FindStudentByParam();

if (indexes.size() == 1 && indexes[0] == -1)

break; //если был выбран выход

for (int i = 0; i < indexes.size(); i++) {

int index = indexes[i];

if (CheckIsManipulatingAdmin(studentsArray[index])) {

WaitEnterInput();

break;

}

do {

ClearTerminal();

HeaderSecondLevel(HEADER);

StudentWorkCourseTable(studentsArray[index]);

}while (SelectMark(index));

}

} while (true);

ClearTerminal();

StudentFileRewrite();

LoadStudentsFromFile();

}

void SetStudentCourseTheme() {

const string HEADER = "МЕНЮ УСТАНОВКИ ТЕМЫ РАБОТЫ СТУДЕНТА";

HeaderSecondLevel(HEADER);

vector<int> indexes;

do {

indexes = FindStudentByParam();

ClearTerminal();

if (indexes.size() == 1 && indexes[0] == -1)

break; //если был выбран выход

HeaderSecondLevel(HEADER);

for (int i = 0; i < indexes.size(); i++) {

int index = indexes[i];

if (CheckIsManipulatingAdmin(studentsArray[index]))

{

WaitEnterInput();

return;

}

StudentWorkCourseTable(studentsArray[index]);

cout << "Вы хотите установить новую тему этому студенту\n";

if (GetUserApprove()) {

studentsArray[index].setCourseWorkTheme();

ClearTerminal();

StudentWorkCourseTable(studentsArray[index]);

WaitEnterInput();

}

ClearTerminal();

HeaderSecondLevel(HEADER);

}

} while (true);

ClearTerminal();

StudentFileRewrite();

LoadStudentsFromFile();

}

//Выдать доступ студенту

void GiveAccesStudents() {

const string HEADER = "МЕНЮ ПРОСМОТРА НЕПОДТВЕРЖДЕННЫХ ПОЛЬЗОВАТЕЛЕЙ";

vector<int> indexesWithoutAcces = GetStudentsWithoutAcces();

vector<int> indexesToDelete;

for (int index : indexesWithoutAcces) {

HeaderSecondLevel(HEADER);

AccoutTable(studentsArray[index]);

cout << "Отклонить или принять заявку? (1- принять, 0 - отклонить)\n";

int choice = GetIntegerInput(0, 1);

if (choice) {

studentsArray[index].setAcces(1);

}

else {

indexesToDelete.push\_back(index);

}

cout << "Запись успешно " << (choice ? "принята" : "отклонена")<<endl;

WaitEnterInput();

}

DeleteStudentsArray(indexesToDelete);

RefreshStudentsId();

StudentFileRewrite();

LoadStudentsFromFile();

string message = "ВСЕ ЗАПИСИ ПРОСМОТРЕНЫ";

LogMessage(message);

WaitEnterInput();

}

//Отобразить на экране информацию о неуспевающих студентах на конкретную дату.

void IndividualTask() {

const string HEADER = "НАЙТИ НЕУСПЕВАЮЩИХ СТУДЕНТОВ";

HeaderSecondLevel(HEADER);

ShowDeadLinesList();

cout << "Выберете дату: (0 - если хотите выйти)" << setw(INPUT\_PADDING) << "";

int input = GetIntegerInput(0, NUM\_OF\_DEADLINES);

if (input == 0) {

ClearTerminal();

return;

}

input--; //декрементируем для индекса

cout << "Неуспевающие студенты по выбранной дате:"<<endl;

for (int i = 0; i < studentsArray.size(); i++) {

if (studentsArray[i].getMark(input) == DEFAULT\_MARK) {

StudentWorkCourseTable(studentsArray[i]);

}

}

WaitEnterInput();

}

//Вывести информацию о текщей учетной записи

void PrintAccountData() {

const string HEADER = "ТЕКУЩАЯ УЧЕТНАЯ ЗАПИСЬ";

HeaderSecondLevel(HEADER);

StudentWorkCourseTable(\*userAccountLink);

WaitEnterInput();

}

void EditAccountData() {

const string HEADER = "ИЗМЕНЕНИЕ ДАННЫХ УЧЕТНОЙ ЗАПИСИ";

HeaderSecondLevel(HEADER);

StudentWorkCourseTable(\*userAccountLink);

userAccountLink->StudentEdit();

StudentFileRewrite();

WaitEnterInput();

}

//Предоставить студенту доступ к редактированию ссылки

void SetCourseWorkLink() {

const string HEADER = "РЕДАКТИРОВАНИЕ ССЫЛКИ КУРСОВОЙ РАБОТЫ";

HeaderSecondLevel(HEADER);

cout << "Текущая ссылка на курсовую работу:\n"<< userAccountLink->getCourseWorkLink()<<endl;

cout << "Вы хотите изменить ссылку\n";

if (!GetUserApprove())

return;

string link = GetLinkInput("Введите новую ссылку:");

userAccountLink->SetCourseWorkStorageLink(link);

string successMessage = "Ссылка успешно добавлена";

LogMessage(successMessage);

WaitEnterInput();

StudentFileRewrite();

}

void GetStudentByParam() {

const string HEADER = "ПОИСК СТУДЕНТА";

HeaderSecondLevel(HEADER);

vector<int> indexes = FindStudentByParam();

if (indexes.size() == 1 && indexes[0] == -1)

return; //если был выбран выход

for (int i : indexes) {

if(studentsArray[i].getId()!=1)

StudentWorkCourseTable(studentsArray[i]);

}

WaitEnterInput();

}