**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение**

**высшего образования**

**«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**ОТЧЁТ ПО ДИСЦИПЛИНЕ**

**«**Автоматизация процессов жизненного цикла программных средств**»**

**На тему:** «Приложение по учету успеваемости студентов»

**Куратор проекта:** \_\_\_\_\_\_\_\_ / Поляков Владимир Анатольевич / 191-351

**Студент:** \_\_\_\_\_\_\_\_ / Герасименко Демид Андреевич, 191-351 /

**Студент:** \_\_\_\_\_\_\_\_/ Щеголькова Василиса Васильевна, 191-351 /

Москва 2021

**СОДЕРЖАНИЕ**

[**1** **СТРУКТУРА ПРОЕКТА** 5](#_Toc89197265)

[**1.1 Концептуальная модель предметной области** 5](#_Toc89197266)

[**1.2 Входные данные** 5](#_Toc89197267)

[**1.3 Выходные данные** 5](#_Toc89197268)

[**1.4 Спецификация обработки информации** 6](#_Toc89197269)

[**1.4.1 Добавление студента** 6](#_Toc89197270)

[**1.4.2 Редактирование данных о студенте** 6](#_Toc89197271)

[**1.4.3 Добавление задания** 6](#_Toc89197272)

[**1.4.4 Редактирование данных о задании** 7](#_Toc89197273)

[**1.4.5 Добавление группы** 7](#_Toc89197274)

[**1.4.6 Редактирование данных о группе** 8](#_Toc89197275)

[**1.4.7 Добавление дисциплины** 8](#_Toc89197276)

[**1.4.8 Редактирование данных о дисциплине** 8](#_Toc89197277)

[**1.4.9 Добавление занятия** 8](#_Toc89197278)

[**1.4.10 Выставление оценки** 9](#_Toc89197279)

[**1.4.11 Анализ студентов** 9](#_Toc89197280)

[**1.4.12 Формализация критериев поиска студентов с задолженностями** 10](#_Toc89197281)

[**1.4.13 Формализация критериев поиска наиболее успевающих студентов** 11](#_Toc89197282)

[**1.4.14 Формализация критериев сортировки студентов по количеству задолженностей, количеству предпринятых попыток сдачи и по преуспевающим программу студентам** 11](#_Toc89197283)

[**1.4.15 Формализация критериев сортировки задолженностей конкретного студента** 11](#_Toc89197284)

[**2 РАБОЧИЙ ПРОЕКТ** 13](#_Toc89197285)

[**2.1 Общие сведения о работе системы** 13](#_Toc89197286)

[**2.2 Требования к техническому и программному обеспечению** 13](#_Toc89197287)

[**2.3 Функциональное назначение** 13](#_Toc89197288)

[**2.4 Общий алгоритм программного продукта** 13](#_Toc89197289)

[**2.5 Инсталляция и выполнение программного продукта** 14](#_Toc89197290)

[**2.6 Общий алгоритм программного продукта** 14](#_Toc89197291)

[**2.6.1 Структура программного проекта** 14](#_Toc89197292)

[**2.6.2 Функциональное назначение модулей, классов, объектов, методов** 16](#_Toc89197293)

[**2.7 Руководство пользователя** 26](#_Toc89197294)

[**2.7.1 Старт (главное меню)** 27](#_Toc89197295)

[**2.7.2 Работа с таблицами** 28](#_Toc89197296)

[**2.8 Сообщения системы** 42](#_Toc89197297)

[**2.8.1 Незаполненные поля** 42](#_Toc89197298)

[**2.8.2 Изменение группы у дисциплины** 42](#_Toc89197299)

[**2.8.3 Существующее название добавляемой группы** 43](#_Toc89197300)

[**2.8.4 Изменение группы у студента** 43](#_Toc89197301)

[**2.8.5 Некорректная дата у задания** 44](#_Toc89197302)

[**2.8.6 Некорректный максимальный балл у задания** 44](#_Toc89197303)

[**2.8.7 Существующий порядковый номер у добавляемого задания** 45](#_Toc89197304)

[**2.8.8 Поздняя дата проведения текущего занятия** 45](#_Toc89197305)

[**2.8.9 Пустое поле оценки** 45](#_Toc89197306)

[**2.8.10 Оценка превышает максимальную оценку за задание** 46](#_Toc89197307)

[**2.8.11 Неудовлетворительная оценка у студента** 46](#_Toc89197308)

# **СТРУКТУРА ПРОЕКТА**

**1.1 Концептуальная модель предметной области**

По результатам анализа предметной области была разработана инфологическая модель задачи, представленная на соответствующей диаграмме классов на рисунке Б.1 в приложении Б. Модель содержит классы «Группа», «Студент», «Задание» и «Оценка», и представляющие объекты предметной области.

Задание характеризуется идентификационным номером, дисциплиной, видом занятия темой занятия, порядковым номером темы по плану занятий, сроком выполнения задания (датой), номером варианта, количеством предпринятых студентом попыток и оценочным баллом. Задание выдается всем студентам группы.

Группа характеризуется идентификационным номером, названием группы, специальностью, курсом. У каждой группы есть свои дисциплины, по которым студентам выдают задания. Также у каждой группы есть свой список студентов, которым выдаются эти задания.

Студент характеризуется такими атрибутами, как идентификационный номер, фамилия, имя, отчество и номер его учебной группы. Каждому студенту присваивается уникальный идентификационный номер (номер зачетной книжки), который позволяет различать студентов, имеющих одинаковые фамилию и имя.

За выполненное задание студент получает оценку. Оценка характеризуется следующими атрибутами: идентификационный номер, студент, которому выставляется оценка, задание, по которому выставляется оценка, номер варианта задания, количество попыток отчитать задание, статус (выполнено или не выполнено), дата сдачи и оценка.

**1.2 Входные данные**

Входными данными проекта являются:

* данные о группе – название, специальность, курс, факультет;
* данные о студенте – фамилия, имя, отчество, группа;
* данные дисциплины – название, группа;
* данные задания – дисциплина, название, срок, тип задания, тема задания, порядковый номер темы, максимальный балл;
* данные оценки – студент, задание, номер варианта, количество попыток, статус, оценка;
* данные занятия – вид занятия, дата и время занятия, группа, дисциплина.

**1.3 Выходные данные**

Выходными данными проекта являются:

* отсортированный список студентов одной группы, составленный для текущего занятия;
* отсортированный список студентов с задолженностями одной группы, составленный для дополнительного занятия;
* индивидуальный список задолженностей для каждого студента, составленный по определенному критерию (первыми идут те задолженности, у которых срок сдачи истекает (истек) раньше всех) для дистанционного занятия или для редактирования успеваемости вручную;
* список студентов, которых можно поощрить;

### **1.4 Спецификация обработки информации**

**1.4.1 Добавление студента**

На рисунке В.1 в приложении В представлена диаграмма деятельности, детализирующая операцию добавления студента.

При добавлении студента в список пользователю необходимо заполнить поля, содержащие данные о нем.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей, таких как фамилия и имя.

В случае корректности ввода данных система производит запрос менеджеру на генерацию уникального идентификационного номера ID для добавляемого студента, затем создает студента и добавляет его в общий список. Таким образом, система должна всегда при добавлении нового студента контролировать присвоение индивидуального номера (ID).

В случае некорректности ввода данных о студенте, система выдает сообщение об ошибке, пользователю будет предложено ввести данные в обязательные поля.

**1.4.2 Редактирование данных о студенте**

На рисунке В.2 в приложении В представлена диаграмма деятельности, детализирующая операцию редактирования данных о студенте.

При выполнении данного действия пользователю необходимо сначала выбрать студента, затем изменить поля, содержащие данные о студенте.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей, таких как фамилия и имя.

Если данные введены корректно, система изменяет данные о студенте. Иначе система выдает сообщение об ошибке, пользователю будет предложено изменить данные в обязательных полях.

**1.4.3 Добавление задания**

На рисунке В.3 в приложении В представлена диаграмма деятельности, детализирующая операцию добавления задания.

При выполнении данного действия пользователю необходимо заполнить поля, содержащие данные о задании.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей, таких как название, срок, тип задания, тема задания, порядковый номер темы и максимальный балл.

В случае корректности ввода данных система производит запрос менеджеру на генерацию уникального идентификационного номера ID для добавляемого задания, затем создает задание и добавляет его в общий список. Таким образом, система должна всегда при добавлении нового задания контролировать присвоение индивидуального номера (ID).

В случае некорректности ввода данных о задании, система выдает сообщение об ошибке, пользователю будет предложено ввести данные в обязательные поля.

**1.4.4 Редактирование данных о задании**

На рисунке В.4 в приложении В представлена диаграмма деятельности, детализирующая операцию редактирования задания.

При выполнении данного действия пользователю необходимо сначала выбрать задание, затем изменить поля, содержащие данные о нем.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей, таких как название, срок, тип задания, тема задания, порядковый номер темы и максимальный балл.

Если данные введены корректно, система изменит данные о задании. Иначе выдаст сообщение об ошибке, пользователю будет предложено изменить данные в обязательных полях.

**1.4.5 Добавление группы**

На рисунке В.5 в приложении В представлена диаграмма деятельности, детализирующая операцию добавления группы.

При выполнении данного действия пользователю необходимо заполнить поля, содержащие данные о группе.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей, таких как название, специальность, курс и факультет.

В случае корректности ввода данных система производит запрос менеджеру на генерацию уникального идентификационного номера ID для добавляемой группы, затем создает группу и добавляет ее в общий список. Таким образом, система должна всегда при добавлении новой группы контролировать присвоение индивидуального номера (ID).

В случае некорректности ввода данных о группе, система выдает сообщение об ошибке, пользователю будет предложено ввести данные в обязательные поля.

**1.4.6 Редактирование данных о группе**

На рисунке В.6 в приложении В представлена диаграмма деятельности, детализирующая операцию редактирования группы.

При выполнении данного действия пользователю необходимо сначала выбрать группу, затем изменить поля, содержащие данные о ней.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей, таких как название, специальность, курс и факультет.

Если данные введены корректно, система изменит данные о задании. Иначе выдаст сообщение об ошибке, пользователю будет предложено изменить данные в обязательных полях.

**1.4.7 Добавление дисциплины**

На рисунке В.7 в приложении В представлена диаграмма деятельности, детализирующая операцию добавления дисциплины.

При выполнении данного действия пользователю необходимо заполнить поля, содержащие данные о дисциплине.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей, таких как название и группа.

В случае корректности ввода данных система производит запрос менеджеру на генерацию уникального идентификационного номера ID для добавляемой дисциплины, затем создает дисциплину и добавляет ее в общий список. Таким образом, система должна всегда при добавлении новой дисциплины контролировать присвоение индивидуального номера (ID).

В случае некорректности ввода данных о дисциплине, система выдает сообщение об ошибке, пользователю будет предложено ввести данные в обязательные поля.

**1.4.8 Редактирование данных о дисциплине**

На рисунке В.8 в приложении В представлена диаграмма деятельности, детализирующая операцию редактирования дисциплины.

При выполнении данного действия пользователю необходимо сначала выбрать дисциплину, затем изменить поля, содержащие данные о ней.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей, таких как название и группа.

Если данные введены корректно, система изменит данные о дисциплине. Иначе выдаст сообщение об ошибке, пользователю будет предложено изменить данные в обязательных полях.

**1.4.9 Добавление занятия**

На рисунке В.9 в приложении В представлена диаграмма деятельности, детализирующая операцию добавления занятия.

При выполнении данного действия пользователю необходимо сначала выбрать вид занятия, а затем заполнить поля, содержащие данные о нем.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей, таких как вид занятия, дата и время занятия, группа, дисциплина, а также, в зависимости от выбранного вида занятия, следующие поля: номер аудитории, задание, студент, ID конференции и пароль конференции.

В случае корректности ввода данных система производит запрос менеджеру на генерацию уникального идентификационного номера ID для добавляемого занятия, затем создает занятие и добавляет его в общий список. Таким образом, система должна всегда при добавлении нового занятия контролировать присвоение индивидуального номера (ID).

В случае некорректности ввода данных о занятии, система выдает сообщение об ошибке, пользователю будет предложено ввести данные в обязательные поля.

**1.4.10 Выставление оценки**

При добавлении задания система всем студентам, причастным к нему, выставляет нуль-оценку (0 баллов), поэтому выставление оценки, по сути, является ее изменением.

На рисунке В.10 в приложении В представлена диаграмма деятельности, детализирующая операцию изменения оценки.

При выполнении данного действия пользователю необходимо выбрать группу, дисциплину, студента и задание, по которому будет выставляться оценка, а затем изменить соответствующее поле.

После заполнения система проверяет полноту введенных данных, то есть проверяется заполнение обязательных полей и некоторые условия (оценка не может быть больше 100 или меньше 5, или больше максимальной оценки за задание).

Если данные введены корректно, система изменит данные об оценке. Иначе выдаст сообщение об ошибке, пользователю будет предложено изменить данные.

**1.4.11 Анализ студентов**

Анализ студентов проводится с целью формирования ограниченного и упорядоченного по заданным критериям списка. Для ограничения списка на текущих занятиях реализован фильтр «Наиболее успевающие студенты» (у которых сданы все задания в запланированные сроки) и сортировка этого списка по сумме баллов за все работы. Для определения очереди студентов для сдачи текущей работы реализован алгоритм сортировки студентов по количеству задолженностей, количеству попыток сдачи задания и студентам, которые преуспевают по заданиям. Для сдачи задолженностей на дополнительных занятиях реализован фильтр «Студенты с задолженностями», позволяющий посмотреть студентов с задолженностями одной группы. Список «должников» можно отсортировать по количеству задолженностей и количеству предпринятых попыток ликвидировать ту или иную задолженность. Также существует возможность просмотра отсортированного списка задолженностей для конкретного студента на дистанционных занятиях (первыми в списке идут те работы, которые были выданы к выполнению раньше всех).

На рисунке В.11 в приложении В представлена диаграмма деятельности, детализирующая операцию анализа студентов.

Для выполнения анализа необходимо создать одно из 3-х занятий. Таким образом, пользователь может провести анализ студентов с задолженностями или наиболее успевающих студентов.

При создании дополнительного занятия (выборе анализа «Студенты с задолженностями») система осуществляет запрос у менеджера студентов с задолженностями, который выполняет их поиск и возвращает список. Сформированный список путем нажатия на кнопку сортируется по количеству задолженностей и количеству предпринятых попыток сдать ту работу, которая была выбрана, а затем выводится на экран.

При выборе анализа «Наиболее успевающие студенты» система осуществляет запрос у менеджера студентов, у которых все задания сданы в запланированные сроки, который выполняет их поиск и возвращает список. Сформированный список студентов сортируется по сумме баллов за все работы, а затем выводится на экран.

При создании текущего занятия (выборе анализа общего списка студентов для сдачи текущей работы) система производит сортировку по количеству задолженностей, количеству предпринятых попыток отчитать запланированную к сдаче работу (в случае, если количество задолженностей одинаково) и по количеству работ, опережающих план занятий.

При создании дистанционного занятия (выборе анализа задолженностей конкретного студента) система сортирует его задолженности по сроку сдачи, т.е. первыми будут идти те задания, у которых срок сдачи истекает раньше.

**1.4.12 Формализация критериев поиска студентов с задолженностями**

Для проведения анализа необходимо организовать поиск студентов, у которых есть задолженность. Студенты, у которых задолженность есть, характеризуются ненулевым значением количества задолженностей.

Пусть count\_debt – количество задолженностей у рассматриваемого студента.

В соответствии с условиями задачи:

* отсутствие задолженностей у данного студента: count\_debt = 0;
* наличие задолженностей у данного студента: count\_debt > 0.

**1.4.13 Формализация критериев поиска наиболее успевающих студентов**

Для проведения анализа необходимо организовать поиск студентов, у которых нет задолженностей и все работы были сданы в установленные сроки. Такие студенты характеризуются нулевым значением количества задолженностей, дата выполнения задания < срок (дата) выполнения.

Пусть count\_debt – количество задолженностей у рассматриваемого студента, tasks\_on\_time – логическое переменная, характеризующая при значении true выполнение всех выполненных работ в установленные сроки.

В соответствии с условиями задачи:

* отсутствие задолженностей у данного студента: count\_debt = 0;
* наличие свиданий у данного поклонника: count\_debt > 0;
* все выполненные работы выполнены вовремя: tasks\_on\_time = true;
* не все выполненные работы выполнены вовремя: tasks\_on\_time = false.

**1.4.14 Формализация критериев сортировки студентов по количеству задолженностей, количеству предпринятых попыток сдачи и по преуспевающим программу студентам**

Первоначально система производит сравнение студентов по количеству задолженностей, сортируя список в порядке возрастания. В случае совпадения количества задолженностей двух сравниваемых студентов система автоматически переходит к следующему критерию – количество попыток сдачи работы, сортируя список в порядке возрастания. Далее выбираются студенты, которые преуспевают по плану занятий (т.е. которые сдали текущую работу), и перемещаются в конец списка.

Пусть count\_debt – количество задолженностей данного студента, count\_attempts – количество предпринятых попыток сдать данную работу данного студента, task\_completed – логическая переменная, означающая при значении true, что задание выполнено.

Таким образом, в соответствии с условиями задачи, сортировка студентов по количеству задолженностей, количеству предпринятых попыток отчитать работу и количеству преуспевающих работ может производиться по следующим критериям в определенном порядке:

1. count\_debt = min;
2. count\_attemps = min.

В случае, если задание уже выполнено, студент перемещается промежуточный массив, в которым хранятся все преуспевающие студенты. После сортировки списка все студенты из промежуточного массива отправляются в конец общего отсортированного списка, а сам массив уничтожается.

**1.4.15 Формализация критериев сортировки задолженностей конкретного студента**

Для конкретного студента система производит сортировку его задолженностей таким образом, чтобы сроки сдачи заданий окончания были по возрастанию.

Пусть deadline – дата последняя дата сдачи задания.

В соответствии с условиями задачи:

Если (task1.deadline > task2.deadline), то отсортированный список задолженностей будет таким:

1. task2;
2. task1.

**2 РАБОЧИЙ ПРОЕКТ**

**2.1 Общие сведения о работе системы**

Программа предназначена для учета и анализа успеваемости студентов, является приложением, функционирующим под управлением ОС Windows. Хранение данных организованно при помощи файлов формата XML.

Программный продукт разработан в среде Microsoft Visual Studio 2019 на языке программирования C#.

**2.2 Требования к техническому и программному обеспечению**

Разрабатываемая программа должна соответствовать следующим минимальным требованиям, обусловленным заданием к проекту.

Требования к программному обеспечению:

* ОС — Windows 7/8/10;
* Microsoft.NET Framework 4.7.2.

Требования к техническому обеспечению:

* IBM-совместимый компьютер с МП Intel Pentium III или AMD K6 с тактовой частотой 500 МГц и выше;
* не менее 512 МБ ОЗУ;
* НЖМД со свободным пространством 1 МБ.

**2.3 Функциональное назначение**

Разработанный программный продукт предназначен для автоматизации учета успеваемости студентов в университете. Программа может быть использована для учета успеваемости студентов одной учебной группы, т.к. позволяет вводить неограниченное количество групп, дисциплин, студентов, заданий и занятий и выставлять оценки. Также программа позволяет назначать 3 вида занятий: дополнительное, текущее и дистанционное и проводить анализ успеваемости студентов.

**2.4 Общий алгоритм программного продукта**

Разработанная информационная система состоит из двух компонентов: исполняемого модуля «Успеваемость студентов.exe» и папки с базой данных о группах, дисциплинах, студентах, занятиях, заданиях и оценках в формате XML.

После запуска программы можно использовать созданную ранее базу данных. Если база данных отсутствует, программа сама автоматически ее создаст. В нее будут помещены все введенные в программу данные. Далее пользователь может работать с этой папкой базы данных, либо сформировать другую, удалив предыдущую.

Все изменения в базе данных, произведенные в процессе программы, автоматически сохраняются в файлы базы данных после нажатия соответствующих кнопок.

**2.5 Инсталляция и выполнение программного продукта**

Если у программы есть база данных, то необходимо создать папку и поместить туда EXE файл «Успеваемость студентов.exe» и папку «xmls». После этих действий дважды нажать на файл «Успеваемость студентов.exe» для запуска программы.

Если у программы нет базы данных, то необходимо так же создать папку и поместить туда EXE файл «Успеваемость студентов.exe». После двойного нажатия на файл «Успеваемость студентов.exe» программа автоматически создаст папку «xmls» и запустится. Все введенные в программу данные сохранятся в файлы папки «xmls».

**2.6 Общий алгоритм программного продукта**

**2.6.1 Структура программного проекта**

В ходе работы над проектом было принято решение разделить текст программы на отдельные файлы для каждого разработанного класса.

Файлы программного проекта и их назначение представлены в таблице 2.1.

**Таблица 2.1 - Файлы программного проекта**

|  |  |
| --- | --- |
| **Имя файла** | **Назначение** |
| Additional\_Lesson.cs | Содержит реализацию методов класс Additional\_Lesson |
| Current\_Lesson.cs | Содержит реализацию методов класса Current\_Lesson |
| Discipline.cs | Содержит реализацию методов класса Discipline |
| Distance\_Lesson.cs | Содержит реализацию методов класса Distance\_Lesson |
| Group.cs | Содержит реализацию методов класса Group |
| Lesson.cs | Содержит реализацию методов класса Lesson |
| Manager\_Disciplines.cs | Содержит реализацию методов класса Manager\_Disciplines |
| Manager\_Groups.cs | Содержит реализацию методов класса Manager\_Groups |
| Manager\_Lessons.cs | Содержит реализацию методов класса Manager\_Lessons |
| Manager\_Student\_Progress.cs | Содержит реализацию методов класса Manager\_Student\_Progress |
| Manager\_Students.cs | Содержит реализацию методов класса Manager\_Students |

**Продолжение таблицы 2.1**

|  |  |
| --- | --- |
| **Имя файла** | **Назначение** |
| Manager\_Tasks.cs | Содержит реализацию методов класса Manager\_Tasks |
| Student.cs | Содержит реализацию методов класса Student |
| Student\_Progress.cs | Содержит реализацию методов класса Student\_Progress |
| Task.cs | Содержит реализацию методов класса Task |
| fm\_add\_discipline.cs | Содержит реализацию методов класса fm\_add\_discipline |
| fm\_add\_group.cs | Содержит реализацию методов класса fm\_add\_group |
| fm\_add\_student.cs | Содержит реализацию методов класса fm\_add\_student |
| fm\_add\_task.cs | Содержит реализацию методов класса fm\_add\_task |
| fm\_additional\_lesson\_info.cs | Содержит реализацию методов класса fm\_additional\_lesson\_info |
| fm\_create\_lesson.cs | Содержит реализацию методов класса fm\_create\_lesson |
| fm\_current\_lesson\_info.cs | Содержит реализацию методов класса fm\_current\_lesson\_info |
| fm\_disciplines.cs | Содержит реализацию методов класса fm\_disciplines |
| fm\_distance\_lesson\_info.cs | Содержит реализацию методов класса fm\_distance\_lesson\_info |
| fm\_edit\_progress.cs | Содержит реализацию методов класса fm\_edit\_progress |
| fm\_groups.cs | Содержит реализацию методов класса fm\_groups |
| fm\_set\_mark.cs | Содержит реализацию методов класса fm\_set\_mark |

**Продолжение таблицы 2.1**

|  |  |
| --- | --- |
| **Имя файла** | **Назначение** |
| fm\_show\_excellent.cs | Содержит реализацию методов класса fm\_show\_excellent |
| fm\_students.cs | Содержит реализацию методов класса fm\_students |
| fm\_tasks.cs | Содержит реализацию методов класса fm\_tasks |
| StartForm.cs | Содержит реализацию методов класса StartForm |

**2.6.2 Функциональное назначение модулей, классов, объектов, методов**

Реализованные в проекте классы делятся на группы по их назначению:

* Классы, реализующие логику предметной области:
  1. Классы, моделирующие объекты предметной области;
  2. Классы, управляющие согласованностью поведения объектов предметной области;
  3. Вспомогательные классы, реализующие сравнение объектов по различным критериям.
* Классы, реализующие интерфейс взаимодействия с пользователем;

В таблице 2.2 представлены классы, реализующие интерфейс взаимодействия с пользователем.

**Таблица 2.2 – Классы, реализующие интерфейс взаимодействия с пользователем**

|  |  |
| --- | --- |
| **Наименование класса** | **Описание** |
| public partial class fm\_add\_discipline | Класс диалогового окна fm\_add\_discipline, предназначенного для добавления новой дисциплины или изменения уже имеющейся. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_add\_group | Класс диалогового окна fm\_add\_group, предназначенного для добавления новой учебной группы или изменения уже имеющейся. Является наследником от public System::Windows::Forms::Form |

**Продолжение таблицы 2.2**

|  |  |
| --- | --- |
| **Наименование класса** | **Описание** |
| public partial class fm\_add\_student | Класс диалогового окна fm\_add\_student, предназначенного для добавления нового студента или изменения уже имеющегося. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_add\_task | Класс диалогового окна fm\_add\_task, предназначенного для добавления нового задания или изменения уже имеющегося. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_additional\_lesson\_info | Класс диалогового окна fm\_additional\_lesson\_info, предназначенного для отображения параметров дополнительного занятия и очереди студентов для сдачи заданий. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_create\_lesson | Класс диалогового окна fm\_create\_lesson, предназначенного для создания занятия. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_current\_lesson\_info | Класс диалогового окна fm\_current\_lesson\_info, предназначенного для отображения параметров текущего занятия и очереди студентов для сдачи задания. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_disciplines | Класс диалогового окна fm\_disciplines, предназначенного для отображения списка дисциплин. Является наследником от public System::Windows::Forms::Form |

**Продолжение таблицы 2.2**

|  |  |
| --- | --- |
| **Наименование класса** | **Описание** |
| public partial class fm\_current\_lesson\_info | Класс диалогового окна fm\_current\_lesson\_info, предназначенного для отображения параметров текущего занятия и очереди студентов для сдачи задания. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_disciplines | Класс диалогового окна fm\_disciplines, предназначенного для отображения списка дисциплин. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_distance\_lesson\_info | Класс диалогового окна fm\_distance\_lesson\_info, предназначенного для отображения параметров дистанционного занятия и очереди задолженностей студента. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_edit\_progress | Класс диалогового окна fm\_edit\_progress, предназначенного для просмотра и изменения успеваемости студентов. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_groups | Класс диалогового окна fm\_groups, предназначенного для отображения списка учебных групп. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_set\_mark | Класс диалогового окна fm\_set\_mark, предназначенного для выставления оценки студенту за выполненное задание. Является наследником от public System::Windows::Forms::Form |

**Продолжение таблицы 2.2**

|  |  |
| --- | --- |
| **Наименование класса** | **Описание** |
| public partial class fm\_show\_excellent | Класс диалогового окна fm\_show\_excellent, предназначенного для отображения списка наиболее успевающих студентов. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_student\_debts | Класс диалогового окна fm\_student\_debts, предназначенного для отображения упорядоченного списка задолженностей студента. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_students | Класс диалогового окна fm\_students, предназначенного для отображения списка студентов. Является наследником от public System::Windows::Forms::Form |
| public partial class fm\_tasks | Класс диалогового окна fm\_tasks, предназначенного для отображения списка заданий по определенной дисциплине. Является наследником от public System::Windows::Forms::Form |
| public partial class StartForm | Класс диалогового окна StartForm, предназначенного для отображения списка предстоящих занятий и выбора действия. Является наследником от public System::Windows::Forms::Form |

В таблице 2.3 представлены классы, моделирующие объекты предметной области.

**Таблица 2.3 – Классы, моделирующие объекты предметной области**

|  |  |
| --- | --- |
| **Наименование класса** | **Описание** |
| public class Additional\_Lesson | Класс «Дополнительное занятие», предназначенный для хранения сведений о дополнительном занятии. Является наследником класса Lesson |

**Продолжение таблицы 2.3**

|  |  |
| --- | --- |
| **Наименование класса** | **Описание** |
| public class Current\_Lesson | Класс «Текущее занятие», предназначенный для хранения сведений о текущем занятии. Является наследником класса Lesson |
| public class Discipline | Класс «Дисциплина», предназначенный для хранения сведений о дисциплине |
| public class Distance\_Lesson | Класс «Дистанционное занятие», предназначенный для хранения сведения о дистанционном занятии. Является наследником класса Lesson |
| public class Group | Класс «Группа», предназначенный для хранения сведений о группе |
| public class Lesson | Класс «Занятие», предназначенный для хранения сведений о занятии. Является родителем 3-х классов: Additional\_Lesson, Current\_Lesson, Distance\_Lesson |
| public class Student | Класс «Студент», предназначенный для хранения сведений о студенте |
| public class Student\_Progress | Класс «Оценка», предназначенный для хранения о сведениях оценки |
| public class Task | Класс «Задание», предназначенный для хранения сведений о задании |

В таблице 2.4 представлены классы, управляющие согласованностью поведения объектов предметной области.

**Таблица 2.4 – Классы, управляющие согласованностью объектов**

|  |  |
| --- | --- |
| **Наименование класса** | **Описание** |
| public class Manager\_Disciplines | Класс «Менеджер дисциплин», предназначенный для хранения списка дисциплин и управления им |
| public class Manager\_Groups | Класс «Менеджер групп», предназначенный хранения списка групп и управления им |

**Продолжение таблицы 2.4**

|  |  |
| --- | --- |
| **Наименование класса** | **Описание** |
| public class Manager\_Lessons | Класс «Менеджер занятий», предназначенный для хранения списка дополнительных занятий, списка текущих занятий и списка дистанционных занятий и управления ими |
| public class Manager\_Student\_Progress | Класс «Менеджер успеваемости студента», предназначенный для хранения оценок студентов и управления ими |
| public class Manager\_Students | Класс «Менеджер студентов», предназначенный для хранения списка студентов и управления им |
| public class Manager\_Tasks | Класс «Менеджер заданий», предназначенный для хранения списка заданий и управления им |

В приложении Г представлена диаграмма классов, реализующих логику предметной области. На данной диаграмме представлены классы, реализующие объекты предметной области (см. таб. 2.3): управляющие классы (см. таб. 2.4), а также отношения между ними.

Управляющие реализацией логики задачи классы «Manager\_Disciplines», «Manager\_Groups», «Manager\_Lessons», «Manager\_Student\_Progress», «Manager\_Students» и «Manager\_Tasks» контролируют поведение и обеспечивают согласованность состояний объектов соответствующих классов предметной области.

Описание полей, методов и объектов основных классов представлено в таблицах 2.5-2.13.

**Таблица 2.5 — Структура класса Group**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Наименование** | **Тип значения** | **Описание** |
| **Поля** | \_ID | int | Идентификационный номер группы |
| \_name | string | Название группы |
| \_speciality | string | Специальность |
| \_course | int | Курс |
| \_faculty | string | Факультет |
| **Свойства** | Id | int | Идентификационный номер группы |
| Name | string | Название группы |
| Speciality | string | Специальность |
| Course | int | Курс |
| Faculty | string | Факультет |
| **Методы** | Group() | – | Конструктор по умолчанию |
| Group(…) | – | Конструктор инициализации. Параметры – значения полей класса |
| Group(Group) | – | Конструктор копирования |

**Таблица 2.6 – Структура класса Discipline**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Наименование** | **Тип значения** | **Описание** |
| **Поля** | \_ID | int | Идентификационный номер дисциплины |
| \_name | string | Название дисциплины |
| \_group | Group | Группа |
| **Свойства** | Id | int | Идентификационный номер дисциплины |
| Name | string | Название дисциплины |
| Group | Group | Группа |
| **Методы** | Discipline() | – | Конструктор по умолчанию |
| Discipline(…) | – | Конструктор инициализации. Параметры – значения полей класса |
| Discipline(Discipline) | – | Конструктор копирования |

**Таблица 2.7 – Структура класса Student**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Наименование** | **Тип значения** | **Описание** |
| **Поля** | \_ID | string | Идентификационный номер (номер зачетной книжки) |
| \_surname | string | Фамилия |
| \_name | string | Имя |
| \_patronymic | string | Отчество |
| \_group | Group | Группа |
| **Свойства** | Id | string | Идентификационный номер (номер зачетной книжки) |
| Surname | string | Фамилия |
| Name | string | Имя |
| Patronymic | string | Отчество |
| Group | Group | Группа |
| **Методы** | Student() | – | Конструктор по умолчанию |
| Student(…) | – | Конструктор инициализации. Параметры – значения полей класса |
| Student(Student) | – | Конструктор копирования |

**Таблица 2.8 – Структура класса Task**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Наименование** | **Тип значения** | **Описание** |
| **Поля** | \_ID | int | Идентификационный номер задания |
| \_name | string | Название задания |
| \_discipline | Discipline | Дисциплина |
| \_term | DateTime | Срок |
| \_type\_of\_lesson | string | Тип занятия |
| \_theme\_of\_lesson | string | Тема занятия |
| \_serial\_number\_of\_theme | int | Порядковый номер темы |
| \_max\_mark | int | Максимальный балл |
| **Свойства** | Id | int | Идентификационный номер задания |
| Name | string | Название задания |
| Discipline | Discipline | Дисциплина |
| Term | DateTime | Срок |
| Type\_of\_Lesson | string | Тип занятия |
| Theme\_of\_Lesson | string | Тема занятия |
| Serial\_Number\_of\_Theme | int | Порядковый номер темы |
| Max\_Mark | int | Максимальный балл |
| **Методы** | Task() | – | Конструктор по умолчанию |
| Task(…) | – | Конструктор инициализации. Параметры – значения полей класса |
| Task(Task) | – | Конструктор копирования |

**Таблица 2.9 – Структура класса Student\_Progress**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Наименование** | **Тип значения** | **Описание** |
| **Поля** | \_ID | int | Идентификационный номер оценки |
| \_student | Student | Студент |
| \_task | Task | Задание |
| \_variant | int | Вариант |
| \_amount\_of\_attemps | int | Количество попыток сдачи |
| \_completed\_date | DateTime | Дата сдачи |
| \_completed | bool | Статус (сдал/не сдал) |
| \_mark | int | Оценка |
| **Свойства** | Id | int | Идентификационный номер оценки |
| Dtudent | Student | Студент |
| Task | Task | Задание |
| Bariant | int | Вариант |
| Amount\_of\_Attemps | int | Количество попыток сдачи |
| Completed\_Date | DateTime | Дата сдачи |
| Completed | bool | Статус (сдал/не сдал) |
| Mark | int | Оценка |
| **Методы** | Student\_Progress() | – | Конструктор по умолчанию |
| Student\_Progress(…) | – | Конструктор инициализации. Параметры – значения полей класса |
| Student\_Progress(Student\_Progress) | – | Конструктор копирования |

**Таблица 2.10 – Структура класса Lesson**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Наименование** | **Тип значения** | **Описание** |
| **Поля** | \_ID | string | Идентификационный номер занятия |
| \_date\_of\_lesson | DateTime | Дата занятия |
| \_discipline | Discipline | Дисциплина |
| \_group | Group | Группа |
| **Свойства** | Id | string | Идентификационный номер занятия |
| Date\_of\_Lesson | DateTime | Дата занятия |
| Discipline | Discipline | Дисциплина |
| Group | Group | Группа |
| **Методы** | Lesson() | – | Конструктор по умолчанию |
| Lesson(…) | – | Конструктор инициализации. Параметры – значения полей класса |
| Lesson(Lesson) | – | Конструктор копирования |

**Таблица 2.11 – Структура класса Additional\_Lesson: Lesson**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Наименование** | **Тип значения** | **Описание** |
| **Поля** | \_audience\_number | string | Номер аудитории |
| **Свойства** | Audience\_Number | string | Номер аудитории |
| **Методы** | Additional\_Lesson(): base() | – | Конструктор по умолчанию |
| Additional\_Lesson(…): base(…) | – | Конструктор инициализации. Параметры – значения полей класса |
| Additional\_Lesson(Additional\_Lesson): base(Additional\_Lesson) | – | Конструктор копирования |

**Таблица 2.12 – Структура класса Current\_Lesson: Lesson**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Наименование** | **Тип значения** | **Описание** |
| **Поля** | \_audience\_number | string | Номер аудитории |
| \_task | Task | Задание |
| **Свойства** | Audience\_Number | string | Номер аудитории |
| Task | Task | Задание |
| **Методы** | Current\_Lesson(): base() | – | Конструктор по умолчанию |
| Current\_Lesson(…): base(…) | – | Конструктор инициализации. Параметры – значения полей класса |
| Current\_Lesson(Current\_Lesson): base(Current\_Lesson) | – | Конструктор копирования |

**Таблица 2.13 – Структура класса Distance\_Lesson: Lesson**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Наименование** | **Тип значения** | **Описание** |
| **Поля** | \_student | Student | Студент |
| \_conference\_ID | string | Идентификационный номер конференции |
| \_conference\_password | string | Пароль для входа в конференцию |
| **Свойства** | Student | Student | Студент |
| Conference\_Id | string | Идентификационный номер конференции |
| Conference\_Password | string | Пароль для входа в конференцию |
| **Методы** | Distance\_Lesson(): base() | – | Конструктор по умолчанию |
| Distance\_Lesson(…): base(…) | – | Конструктор инициализации. Параметры – значения полей класса |
| Distance\_Lesson(Distance\_Lesson): base(Distance\_Lesson) | – | Конструктор копирования |

**2.7 Руководство пользователя**

Интерфейс программы (рисунок 2.1) достаточно удобен. Программа содержит строку меню, состоящую из одного пункта «Справочники» с выпадающими списком подменю. Кнопки «Создать занятие», «Выданные задание», «Успеваемость студентов», «Наиболее успевающие студенты», «Завершить работу» позволяют производить соответствующие действия над данными.

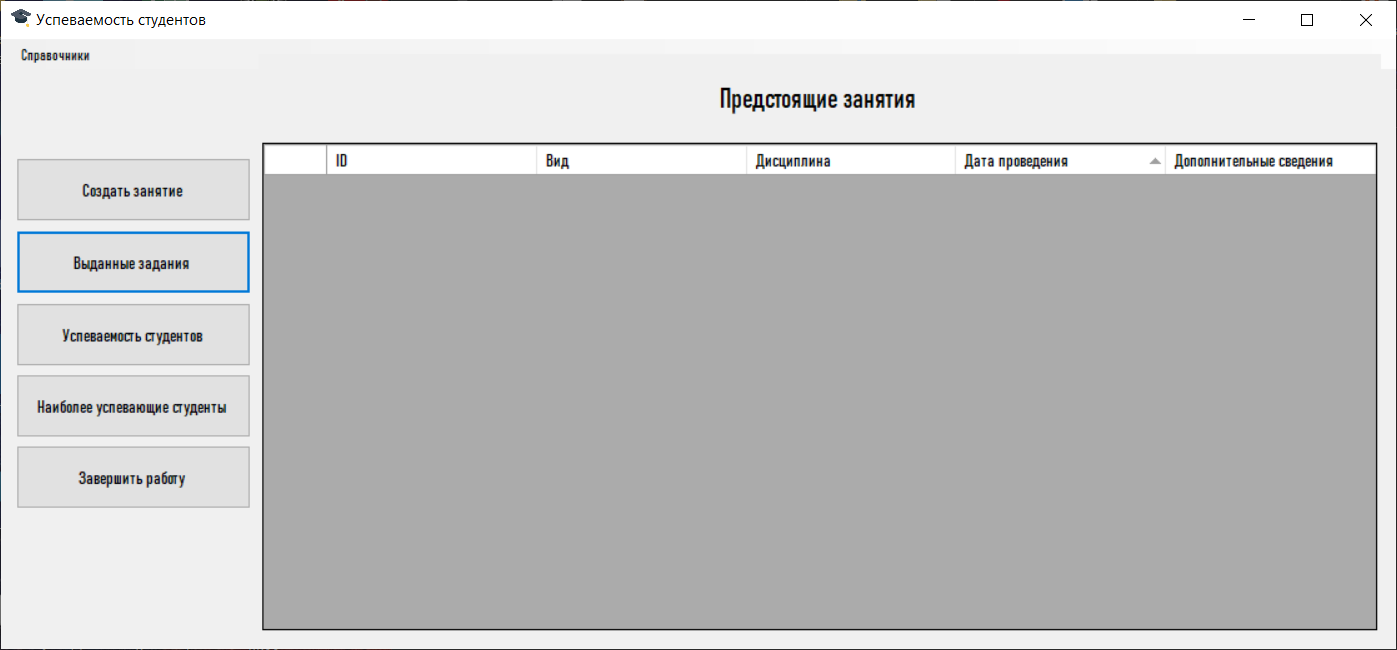


Рисунок 2.1 – Интерфейс программы

Теперь рассмотрим все более подробно.

**2.7.1 Старт (главное меню)**

Начало работы программы сопровождается диалоговым окном «Успеваемость студентов» (рис. 2.2).

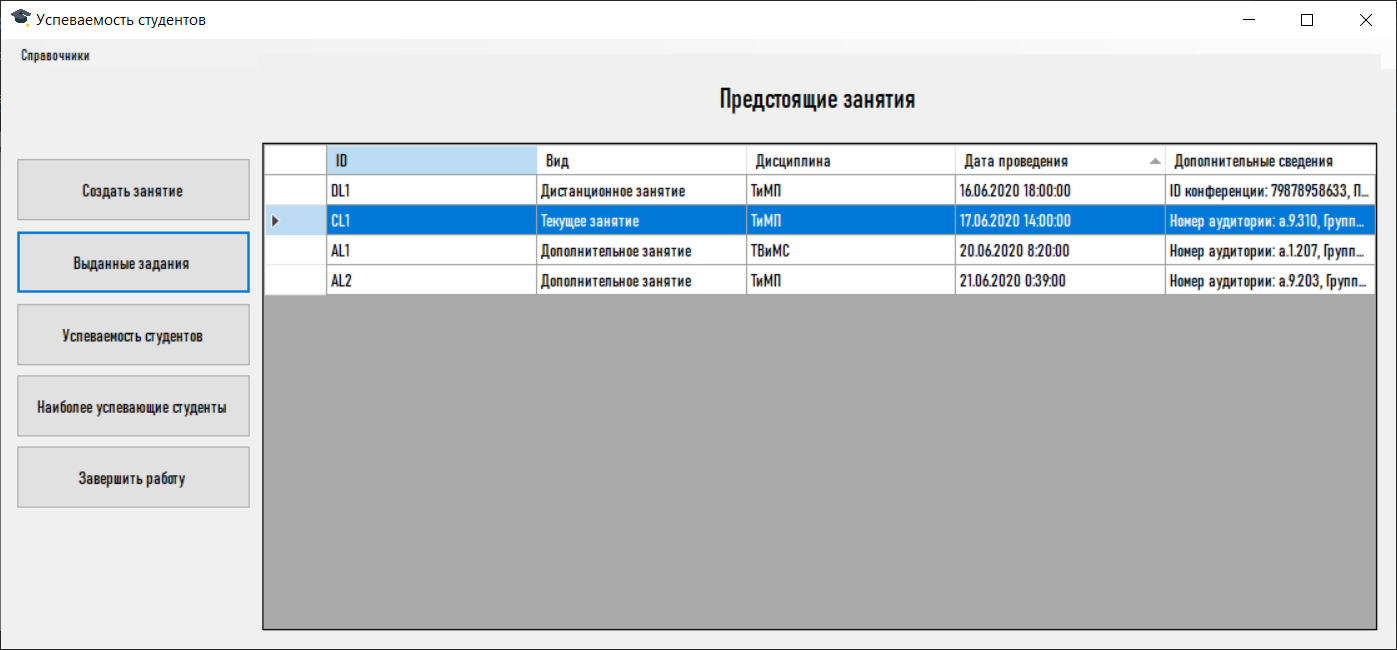


Рисунок 2.2 – Главное меню

Данное диалоговое окно является главным меню программы и обладает следующим перечнем функций:

1. Пункт меню «Справочники» → «Студенты» позволяет посмотреть общий список студентов всех учебных групп.
2. Пункт меню «Справочники» → «Группы» позволяет посмотреть список учебных групп.
3. Пункт меню «Справочники» → «Дисциплины» позволяет посмотреть список дисциплин всех учебных групп.
4. Кнопка «Создать занятие» позволяет создать занятие для защиты заданий студентами.
5. Кнопка «Выданные задания» позволяет посмотреть список заданий конкретной группы по конкретной дисциплине.
6. Кнопка «Успеваемость студентов» позволяет посмотреть успеваемость студента по конкретной дисциплине.
7. Кнопка «Наиболее успевающие студенты» позволяет сформировать список студентов, которых преподаватель может поощрить.
8. Кнопка «Завершить работу» закрывает программу.

**2.7.2 Работа с таблицами**

Чтобы посмотреть список имеющихся студентов, необходимо перейти во вкладку «Справочники» и там нажать на выпадающую вкладку «Студенты» (рис. 2.3).

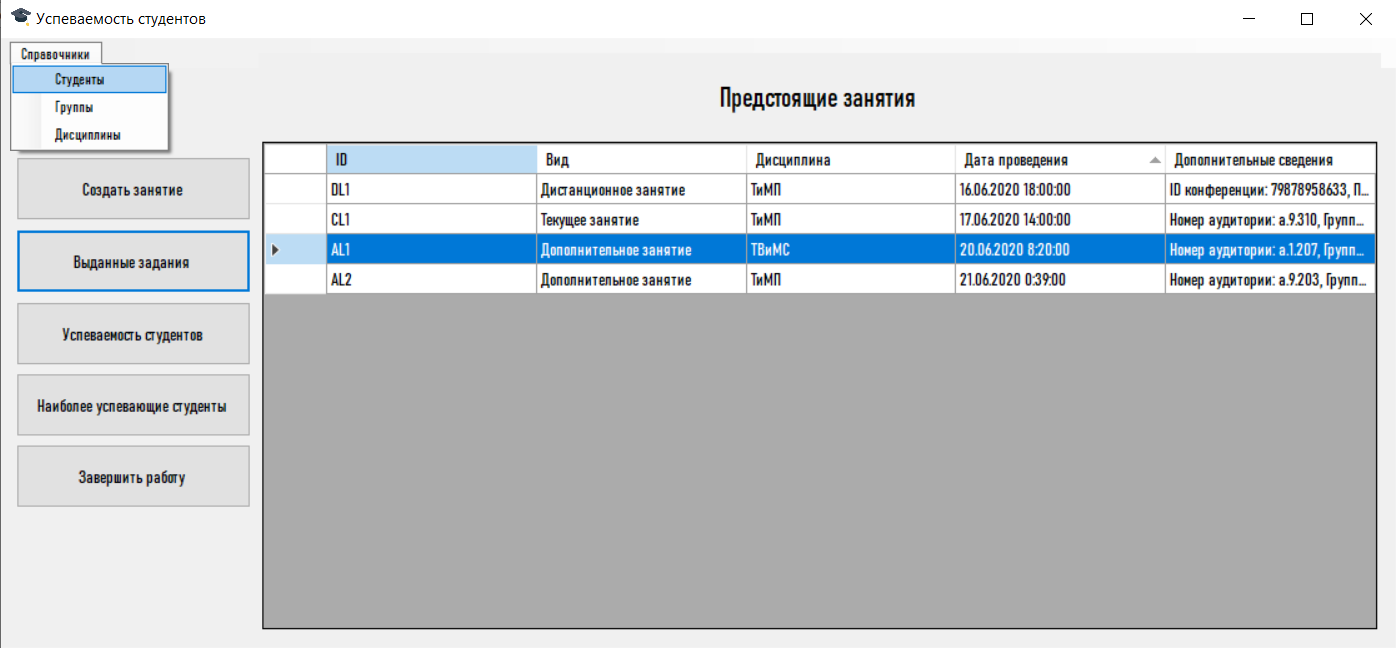


Рисунок 2.3 – Нажатие на вкладку «Студенты»

После осуществленных действий откроется диалоговое окно со списком студентов (рис 2.4).

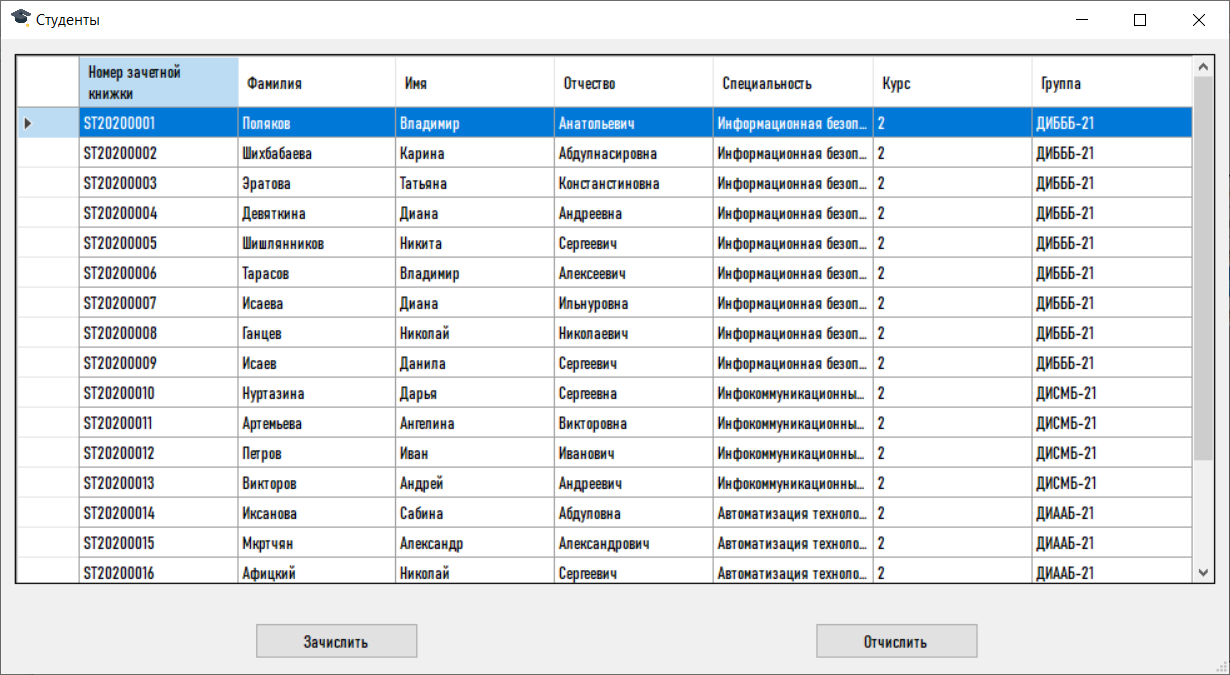


Рисунок 2.4 – Список студентов

Чтобы зачислить студента, необходимо нажать на кнопку «Зачислить». Форма для заполнения данных студента изображена на рис. 2.5.

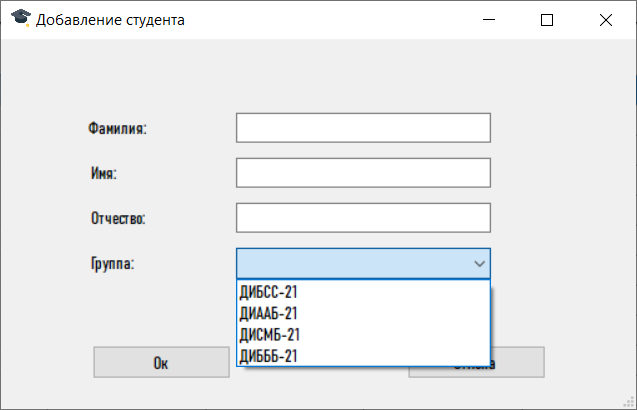


Рисунок 2.5 – Зачисление студента

Для изменения параметров студента необходимо дважды на него нажать левой кнопкой мыши или один раз нажать правой кнопкой мыши и выбрать «Изменить» (рис. 2.6).

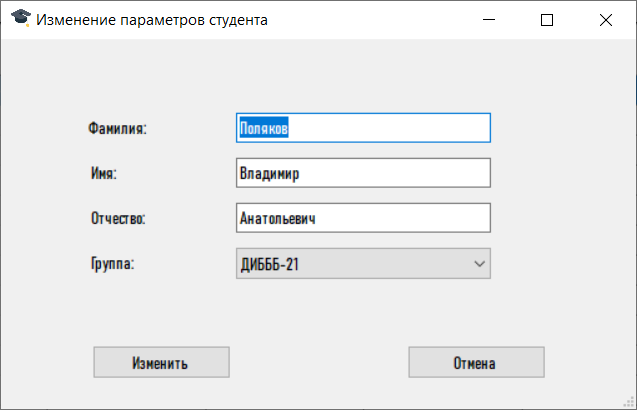


Рисунок 2.6 – Изменение параметров студента

Чтобы посмотреть список имеющихся дисциплин, необходимо перейти во вкладку «Справочники» и там нажать на выпадающую вкладку «Дисциплины». После осуществленных действий откроется диалоговое окно с соответствующим списком (рис. 2.7).

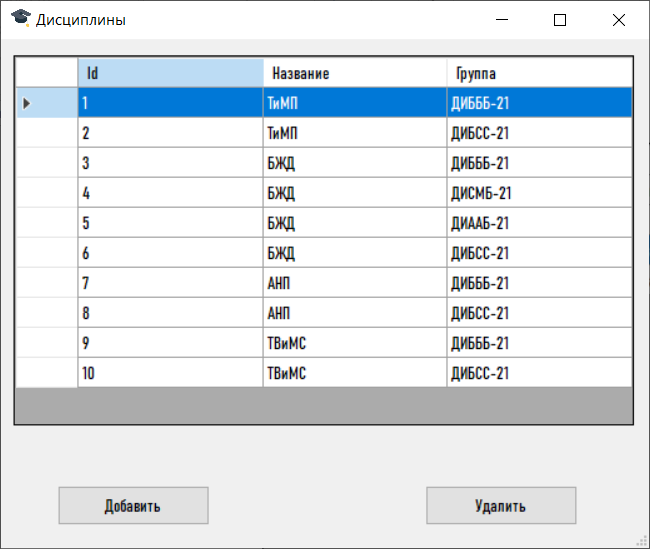


Рисунок 2.7 – Список дисциплин

Чтобы добавить новую дисциплину в учебную группу, необходимо нажать на кнопку «Добавить». Форма для заполнения данных дисциплины изображена на рис. 2.8.

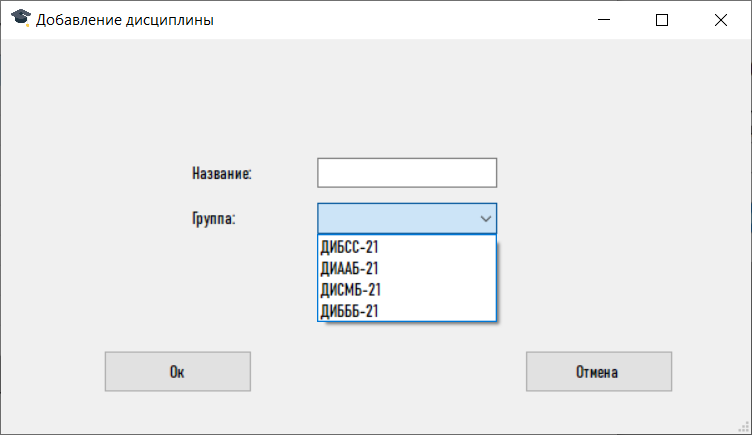


Рисунок 2.8 – Добавление дисциплины

Для изменения параметров студента необходимо дважды на нее нажать левой кнопкой мыши или один раз нажать правой кнопкой мыши и выбрать «Изменить» (рис. 2.9).

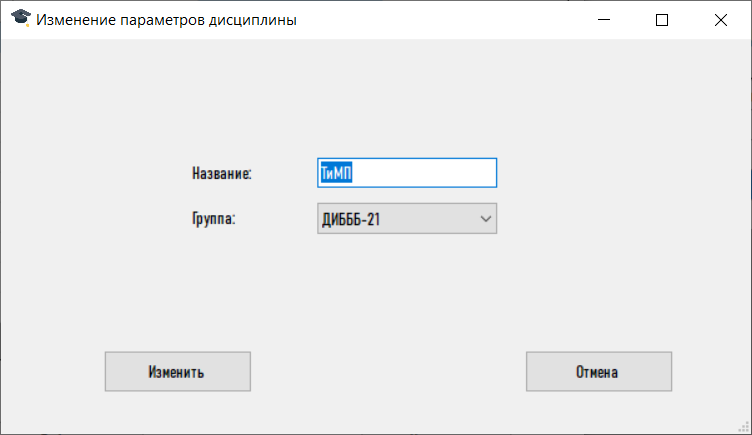


Рисунок 2.9 – Изменение параметров дисциплины

Чтобы посмотреть список имеющихся групп, необходимо перейти во вкладку «Справочники» и там нажать на выпадающую вкладку «Группы». После осуществленных действий откроется диалоговое окно с соответствующим списком (рис. 2.10).

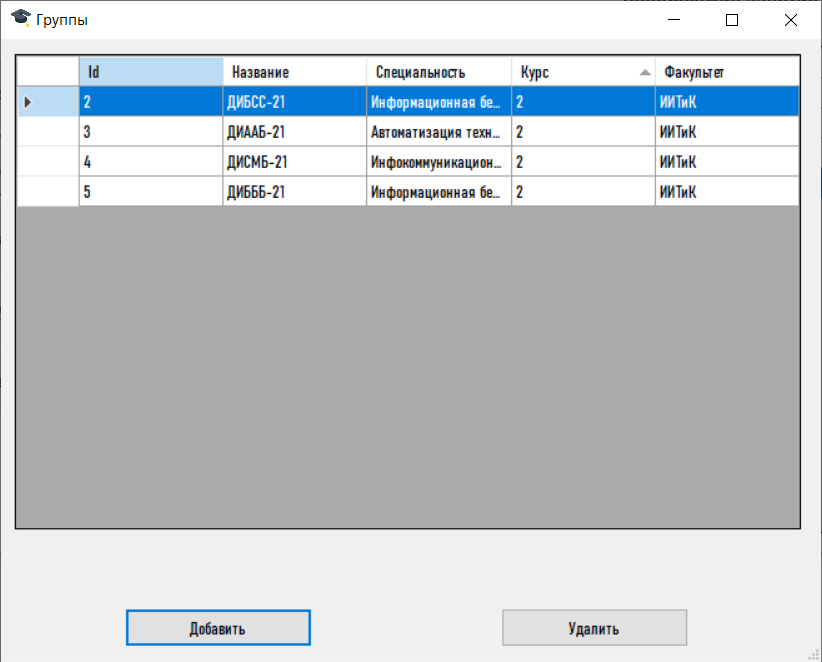


Рисунок 2.10 – Список групп

Чтобы добавить новую группу, необходимо нажать на кнопку «Добавить». Форма для заполнения данных группы изображена на рис. 2.11.

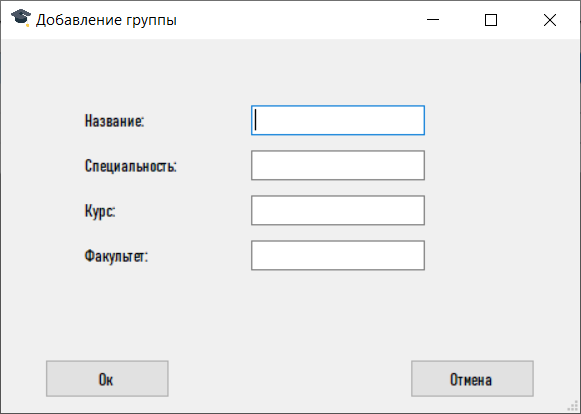


Рисунок 2.11 – Добавление группы

Для изменения параметров группы необходимо дважды на нее нажать левой кнопкой мыши или один раз нажать правой кнопкой мыши и выбрать «Изменить» (рис. 2.12).

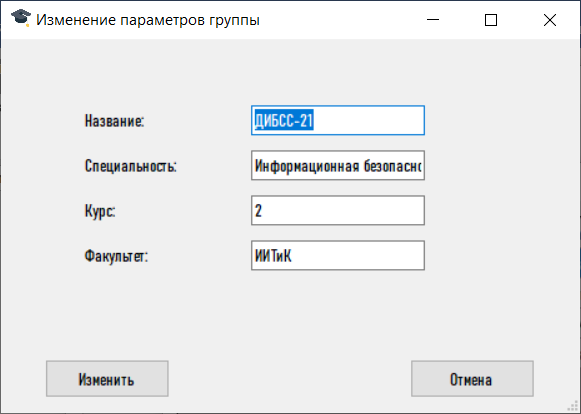


Рисунок 2.12 – Изменение параметров группы

Чтобы назначить занятие, необходимо нажать в главном меню на кнопку «Создать занятие». Программа предлагает 3 вида занятий с разными параметрами. Формы с полями этих занятий изображены на рис. 2.13 – 2.15.

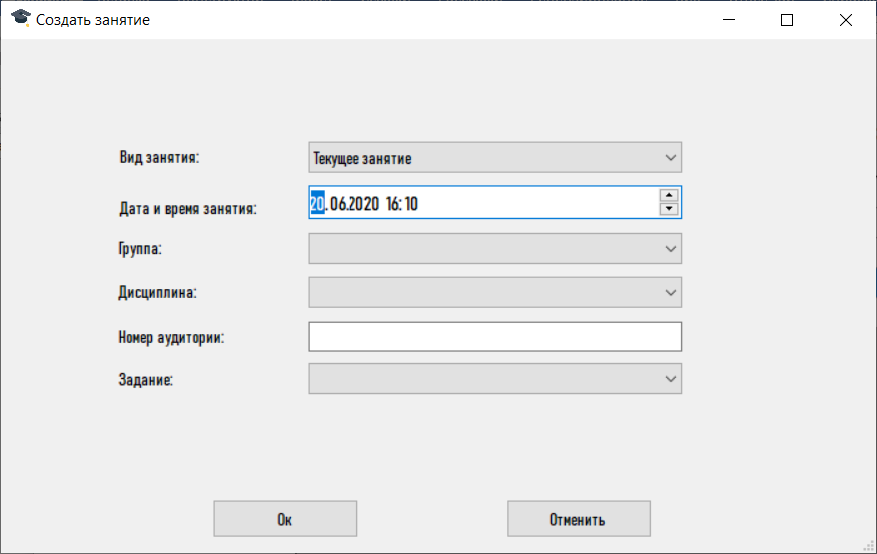


Рисунок 2.13 – Создание текущего занятия

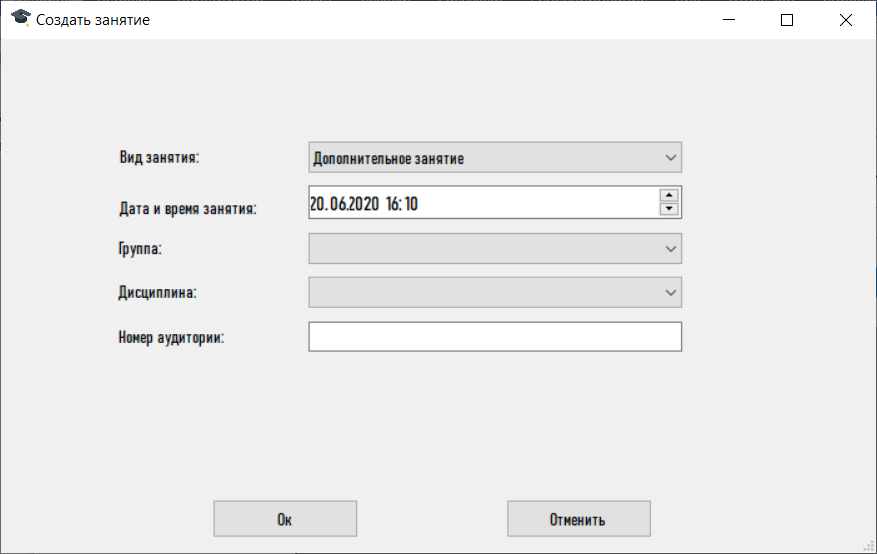


Рисунок 2.14 – Создание дополнительного занятия

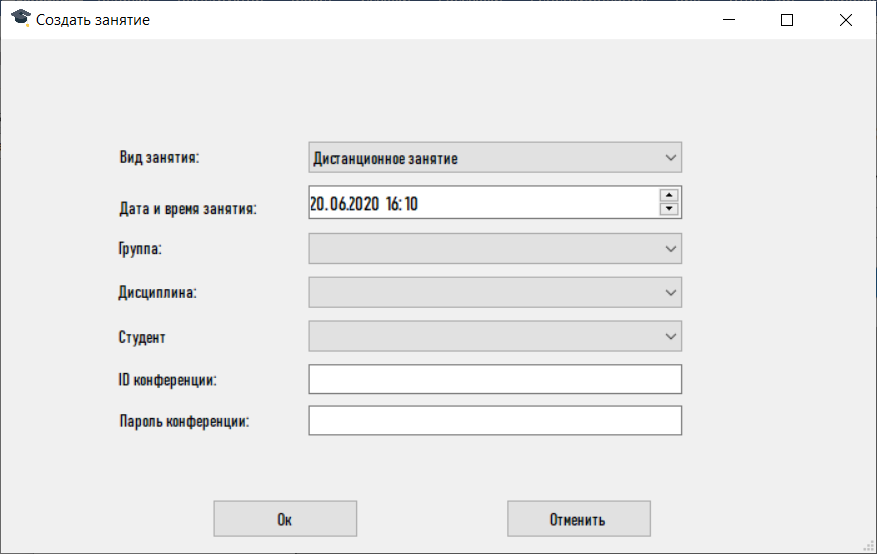


Рисунок 2.15 – Создание дистанционного занятия

Изменение параметров занятий невозможно!

Для просмотра списка выданных заданий, а также их добавления/удаления/изменения, необходимо нажать на кнопку «Выданные задания». Чтобы посмотреть список заданий по конкретной дисциплине конкретной учебной группы, необходимо выбрать дисциплину. Список отобразится автоматически. Диалоговое окно данной кнопки изображено на рис. 2.16.

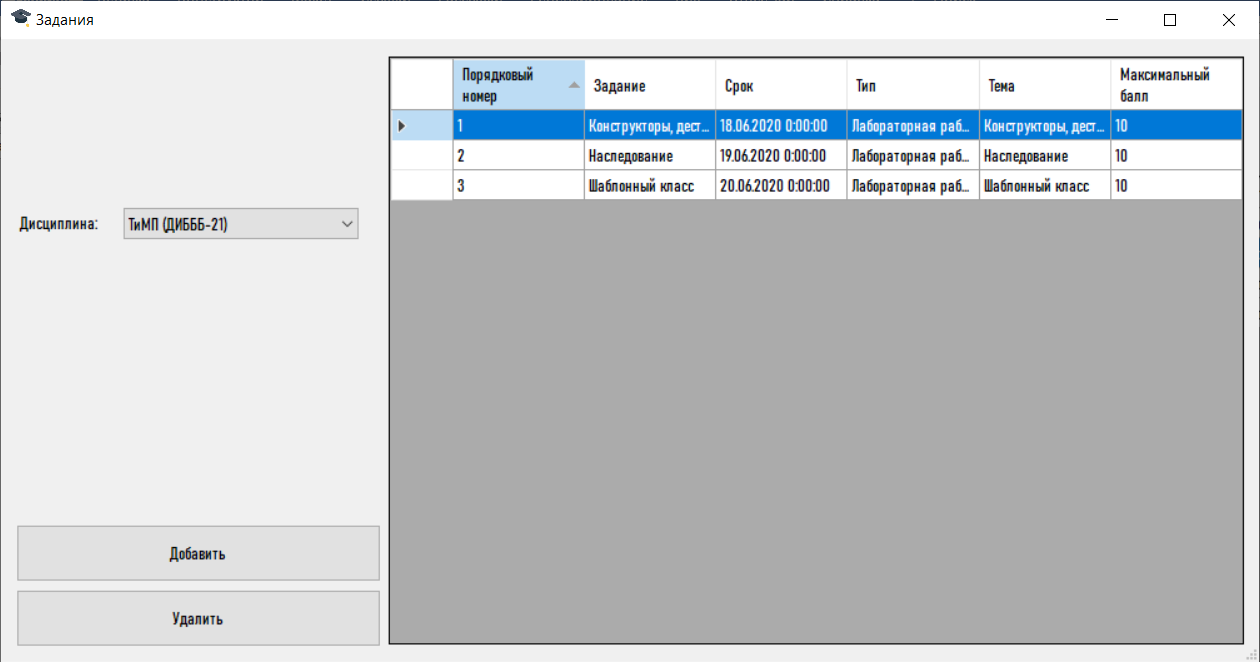


Рисунок 2.16 – Диалоговое окно «Задания»

Чтобы добавить новое задание, необходимо нажать на кнопку «Добавить». Форма для заполнения данных задания изображена на рис. 2.17.

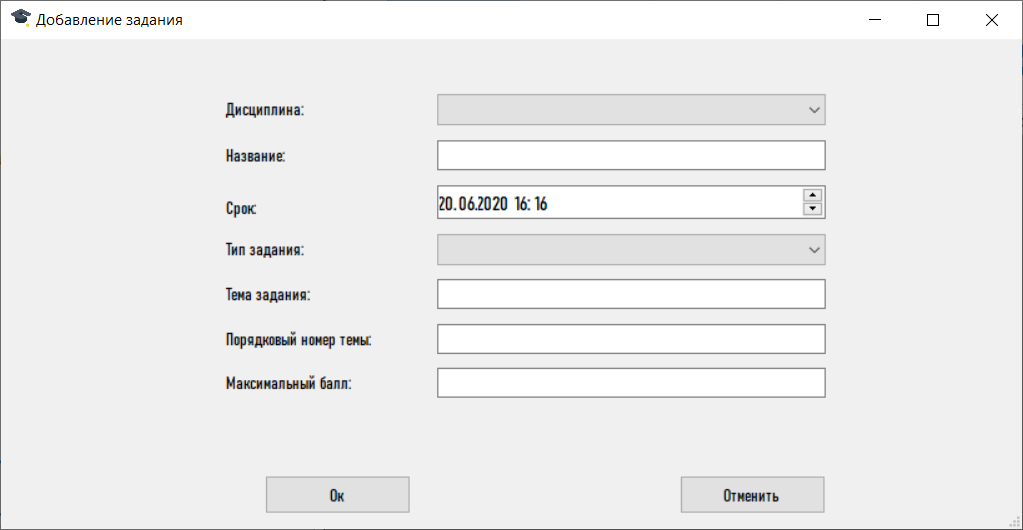


Рисунок 2.17 – Добавление задания

Для изменения параметров заданий необходимо дважды на него нажать левой кнопкой мыши или один раз нажать правой кнопкой мыши и выбрать «Изменить» (рис. 2.18). Изменять можно все параметры, кроме порядкового номера и дисциплины.

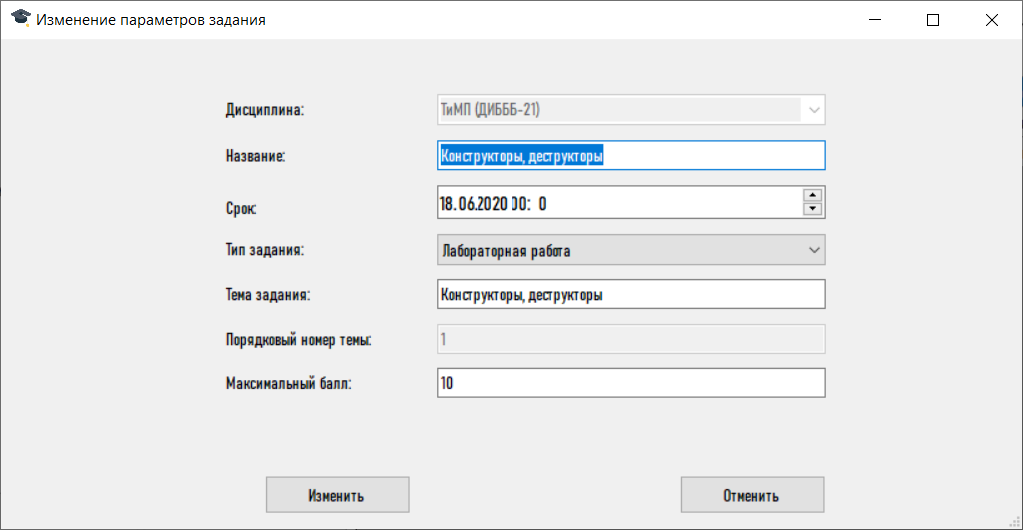


Рисунок 2.18 – Изменение задания

Для просмотра и изменения (вручную) успеваемости студентов конкретной учебной группы по конкретной дисциплине необходимо нажать на кнопку «Успеваемость студентов» и в выходящем диалоговом окне выбрать группу, дисциплину и студента. Диалоговое окно данной кнопки изображено на рис. 2.19.

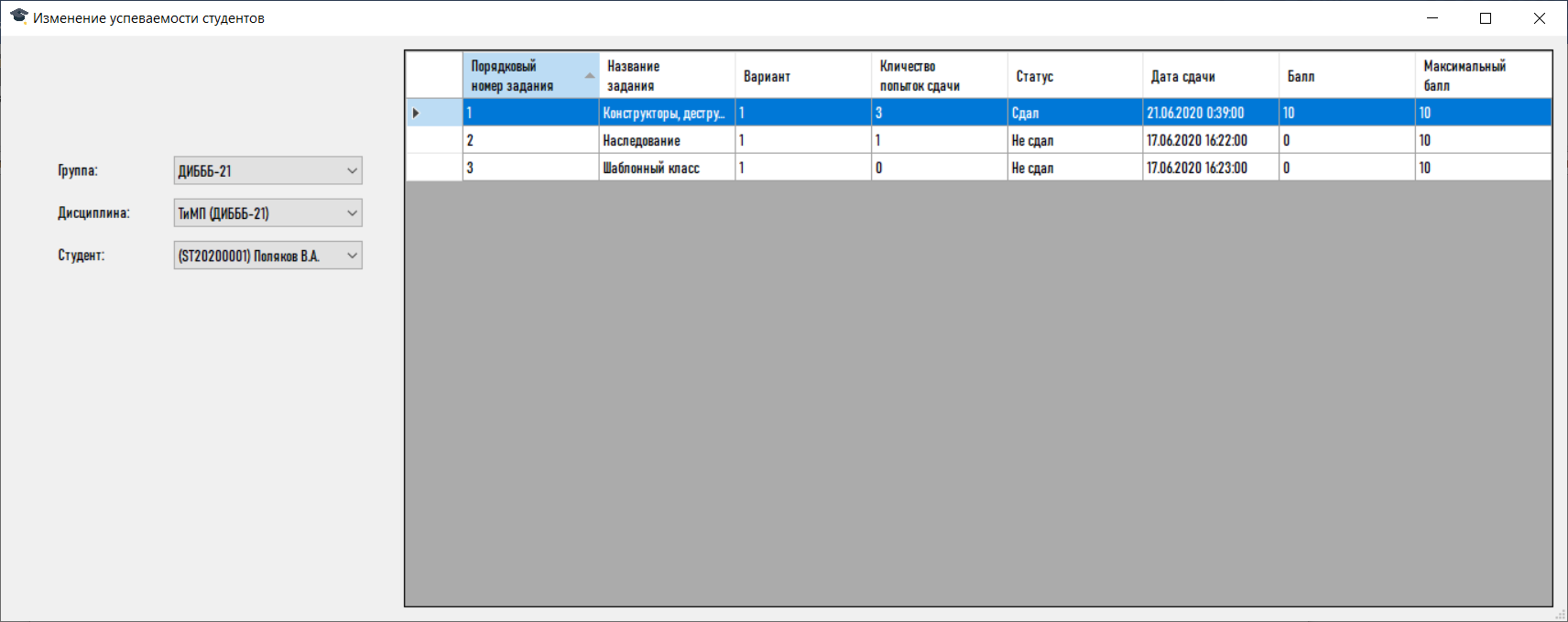


Рисунок 2.19 – Просмотр успеваемости студента

В данном диалоговом окне также можно выставлять (изменять) оценку по заданию. Для этого необходимо по заданию дважды нажать левой кнопкой мыши или один раз правой кнопкой мыши и выбрать «Выставить оценку». Форма для выставления оценки представлена на рис. 2.20.

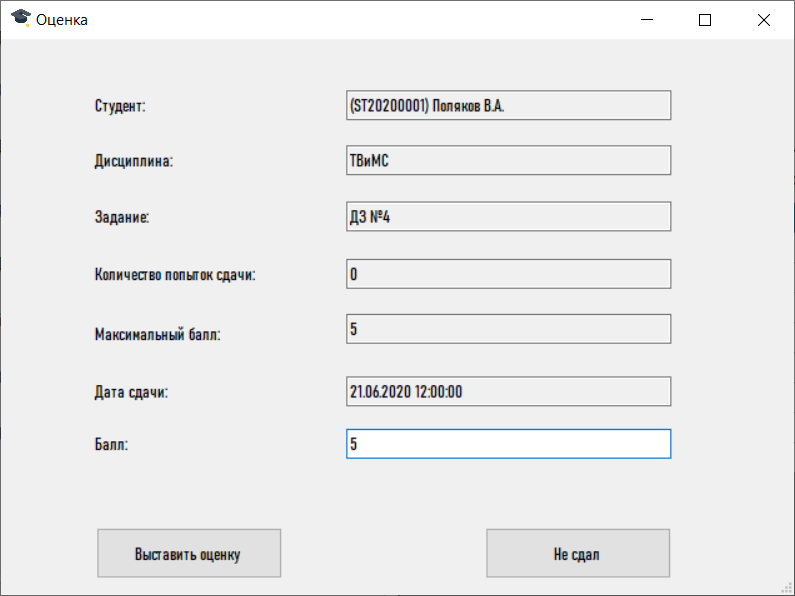


Рисунок 2.20 – Выставление оценки

При нажатии на «Выставить оценку» выставленная в поле «Балл» оценка сохранится, а количество попыток сдать задание увеличится на 1. При нажатии кнопки «Не сдал» балл сохранится тот же, что и был до открытия диалогового окна, а количество попыток сдать задание увеличится на 1.

Для просмотра списка наиболее успевающих студентов, необходимо нажать на кнопку «Наиболее успевающие студенты» выбрать группу и дисциплину, по которой необходимо составить такой список. После этих действий программа автоматически отобразит данный список в упорядоченном формате: первыми будут те, у которых суммарный балл за сдачу всех заданий больше (рис. 2.21).

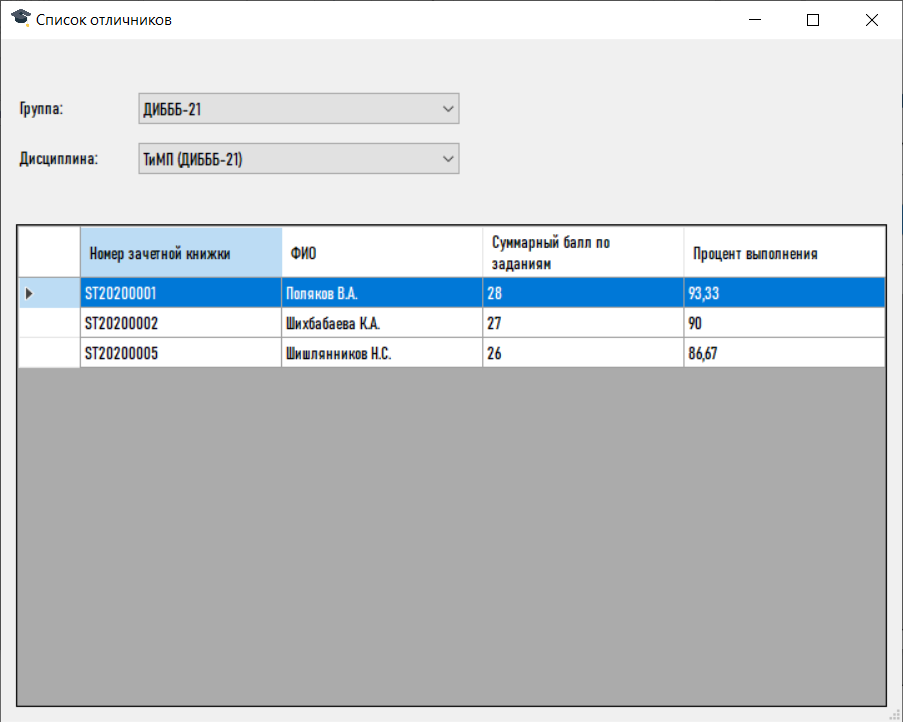


Рисунок 2.21 – Список наиболее успевающих студентов

При нажатии на кнопку «Завершить работу» программа завершит работу, а все введенные в нее ранее данные автоматически сохранятся.

Ранее упоминалось, что программа предполагает наличие 3-х видов занятий. Рассмотрим 1-е из них: текущее занятие. Чтобы выставить оценки по данному занятию, необходимо в главном меню нажать на него дважды левой кнопкой мыши или один раз правой кнопкой мыши и выбрать пункт «Выставить оценки за занятие». После этих действий выйдет диалоговое окно «Информация о текущем занятии» (рис 2.22).

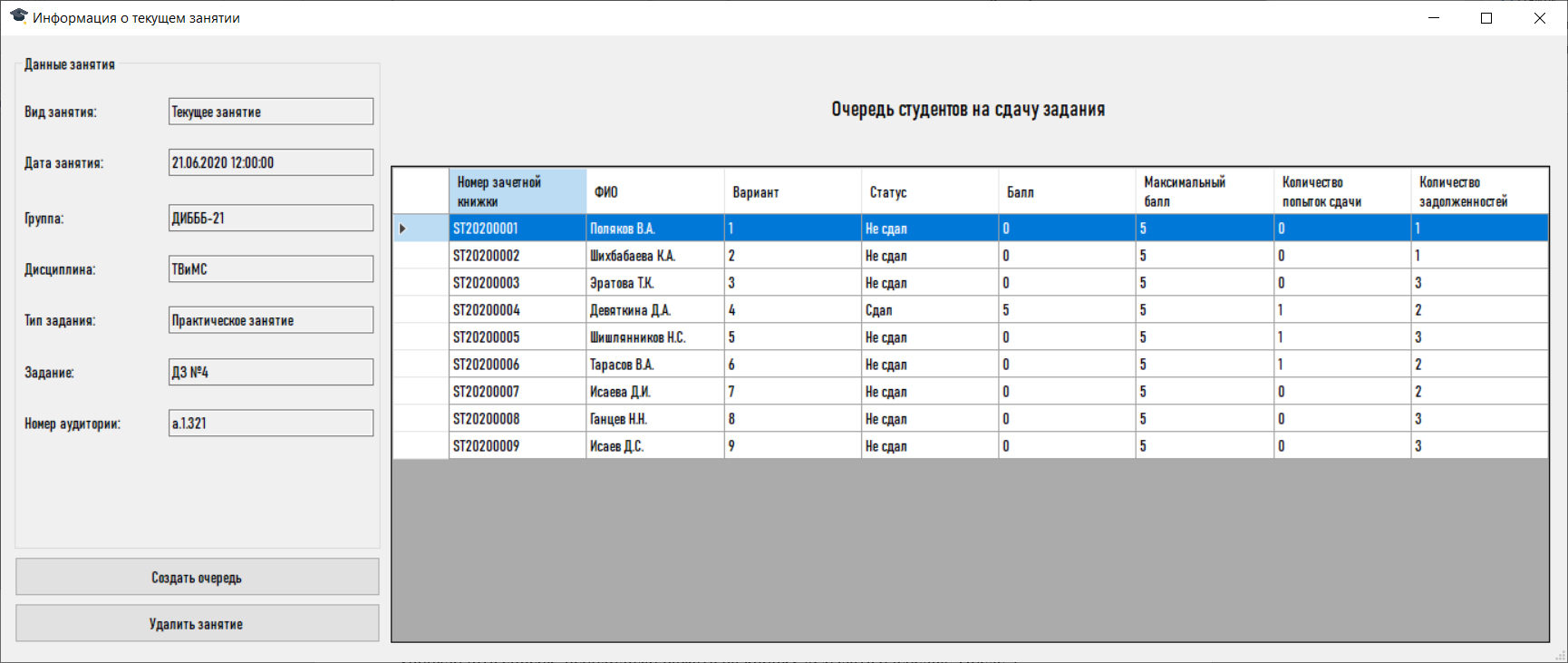


Рисунок 2.22 – Текущее занятие

По заданию курсовой работы необходимо создать очередь для сдачи задания. Очередь создается следующим образом: студенты располагаются в порядке возрастания количества задолженностей; если у двух студентов одинаковое количество задолженностей, то выше будет тот, у кого меньше попыток сдачи этого задания; в последнюю очередь рассматриваются те, кто уже сдал это задание (для сдачи задолженностей или следующих по плану заданий). Чтобы упорядочить список, необходимо нажать на кнопку «Создать очередь». На рис. 2.22 изображен список до нажатия кнопки «Создать очередь», на рис. 2.23 – список после нажатия кнопки «Создать очередь».

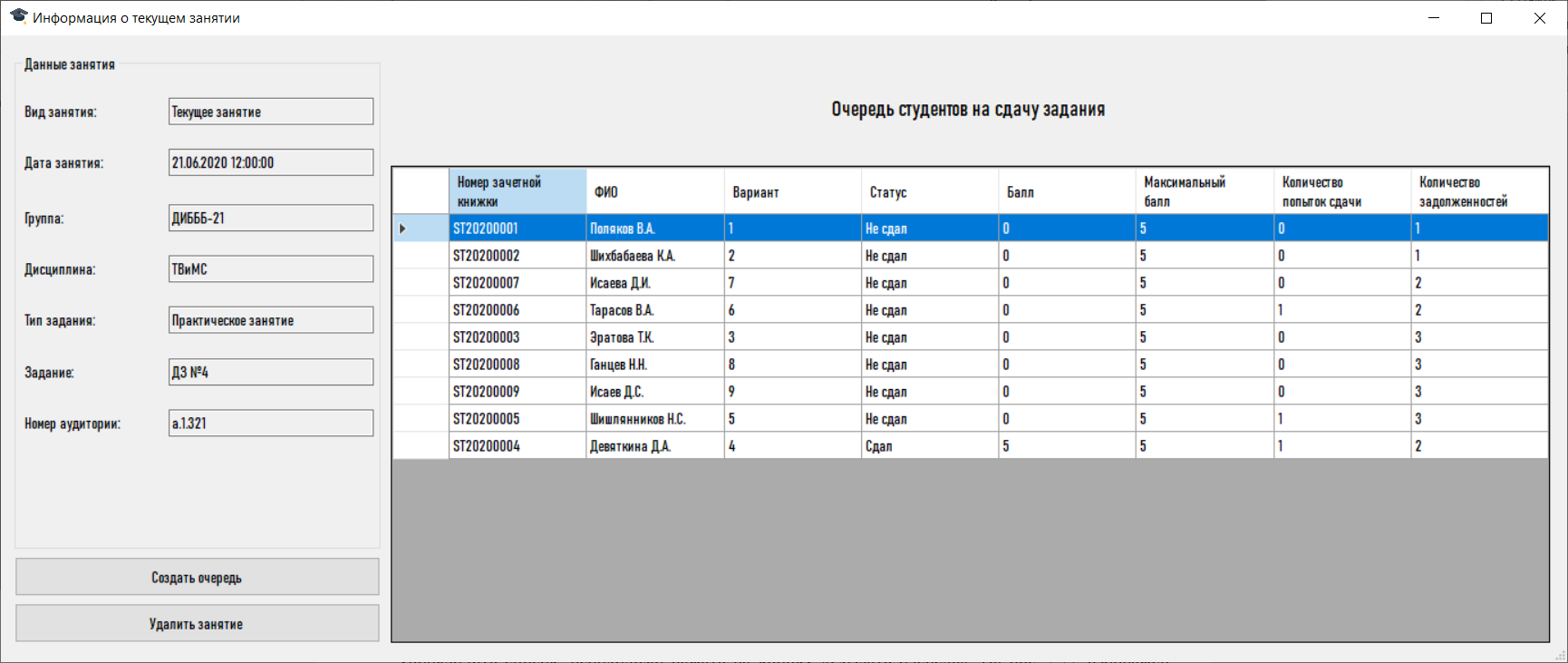


Рисунок 2.23 – Очередь для текущего занятия

Для выставления оценки необходимо нажать дважды левой кнопкой мыши или единожды правой кнопкой мыши по студенту и выбрать «Выставить оценку». Если выбрать в данном меню «Отсутствует», то студент будет удален из списка.

Рассмотрим 2-й вид занятия: дополнительное. Данное занятие проводится по конкретной дисциплине для студентов с задолженностями (задания, которые не сданы и у которых истек срок сдачи). Чтобы выставить оценки по данному занятию, необходимо в главном меню нажать на него дважды левой кнопкой мыши или один раз правой кнопкой мыши и выбрать пункт «Выставить оценки за занятие». После этих действий выйдет диалоговое окно «Информация о дополнительном занятии» (рис 2.24). Для отображения списка задолженностей необходимо в элементе окна выбрать задание, по которому будет создаваться список студентов-должников.

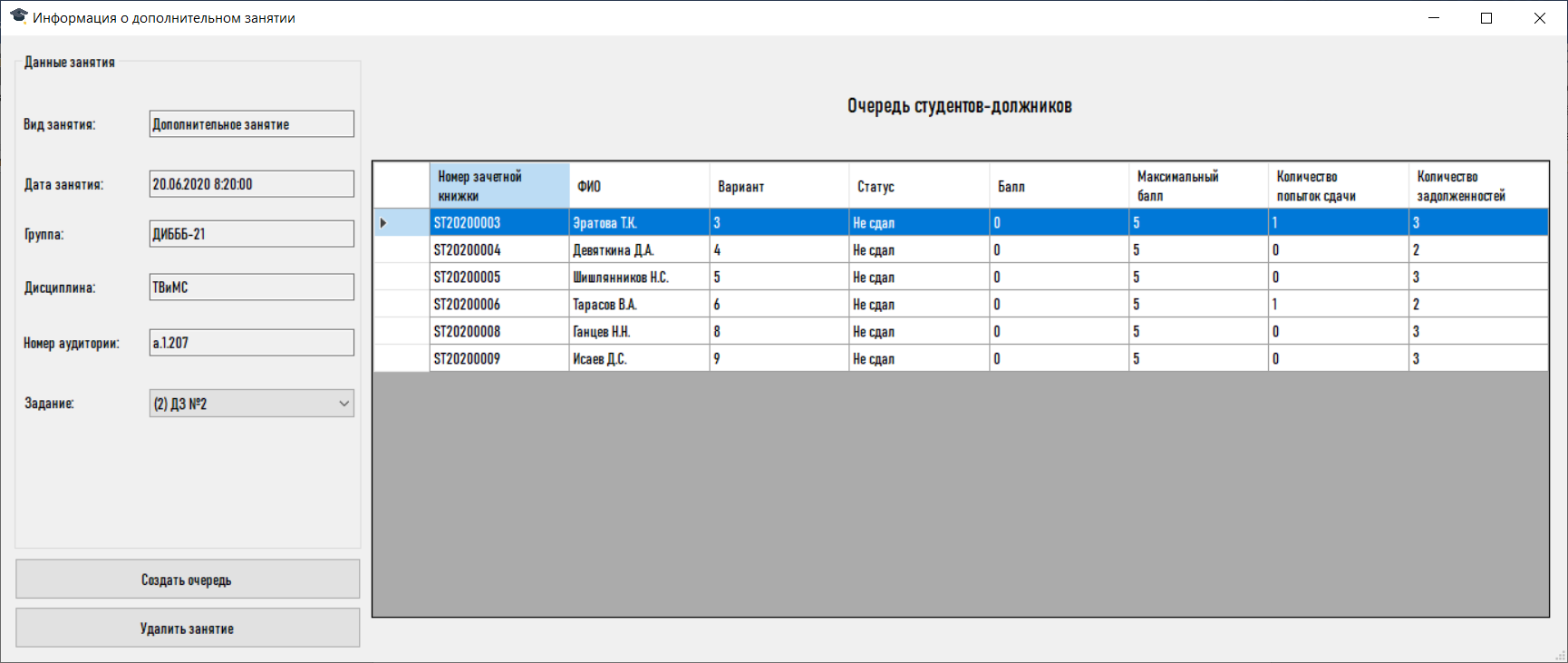


Рисунок 2.24 – Информация о дополнительном занятии

Для данного вида занятия очередь создается следующим образом: студенты располагаются в порядке возрастания количества задолженностей; если у двух студентов одинаковое количество задолженностей, то выше будет тот, у кого меньше попыток сдачи этого задания; в последнюю очередь рассматриваются те, кто уже сдал это задание (для сдачи других задолженностей). Чтобы упорядочить список, необходимо нажать на кнопку «Создать очередь». На рис. 2.24 изображен список до нажатия кнопки «Создать очередь», на рис. 2.25 – список после нажатия кнопки «Создать очередь».

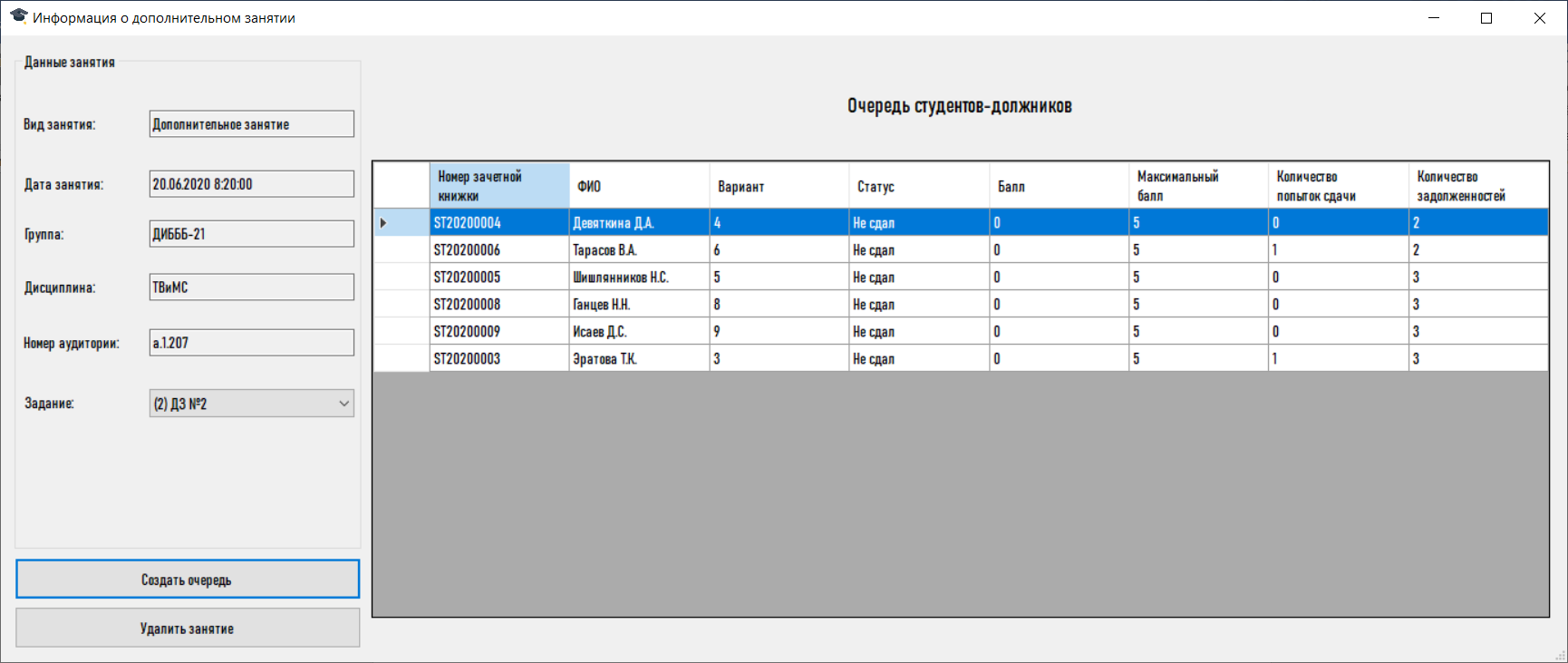


Рисунок 2.25 – Очередь для дополнительного занятия

Для выставления оценки необходимо нажать дважды левой кнопкой мыши или единожды правой кнопкой мыши по студенту и выбрать «Выставить оценку». Если выбрать в данном меню «Отсутствует», то студент будет удален из списка.

Рассмотрим 3-й вид занятия: дистанционное. Данное занятие проводится для одного студента по конкретной дисциплине. Чтобы выставить оценки по данному занятию, необходимо в главном меню нажать на него дважды левой кнопкой мыши или один раз правой кнопкой мыши и выбрать пункт «Выставить оценки за занятие». После этих действий выйдет диалоговое окно «Информация о дистанционном занятии» (рис 2.26).

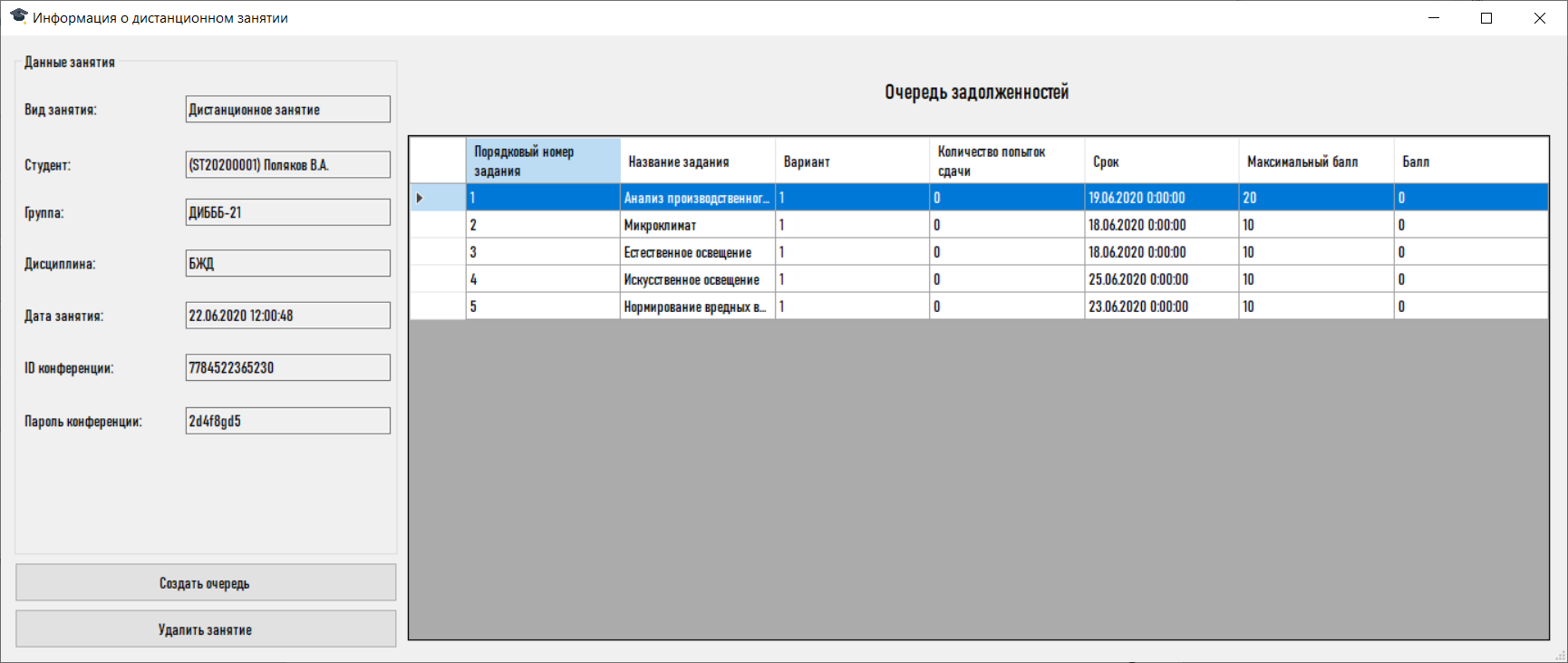


Рисунок 2.26 – Информация о дистанционном занятии

Для данного вида занятия очередь создается следующим образом: первыми идут те, задолженности, у которых срок сдачи истекает (истек) раньше. Чтобы упорядочить список, необходимо нажать на кнопку «Создать очередь». На рис. 2.26 изображен список до нажатия кнопки «Создать очередь», на рис. 2.27 – список после нажатия кнопки «Создать очередь».

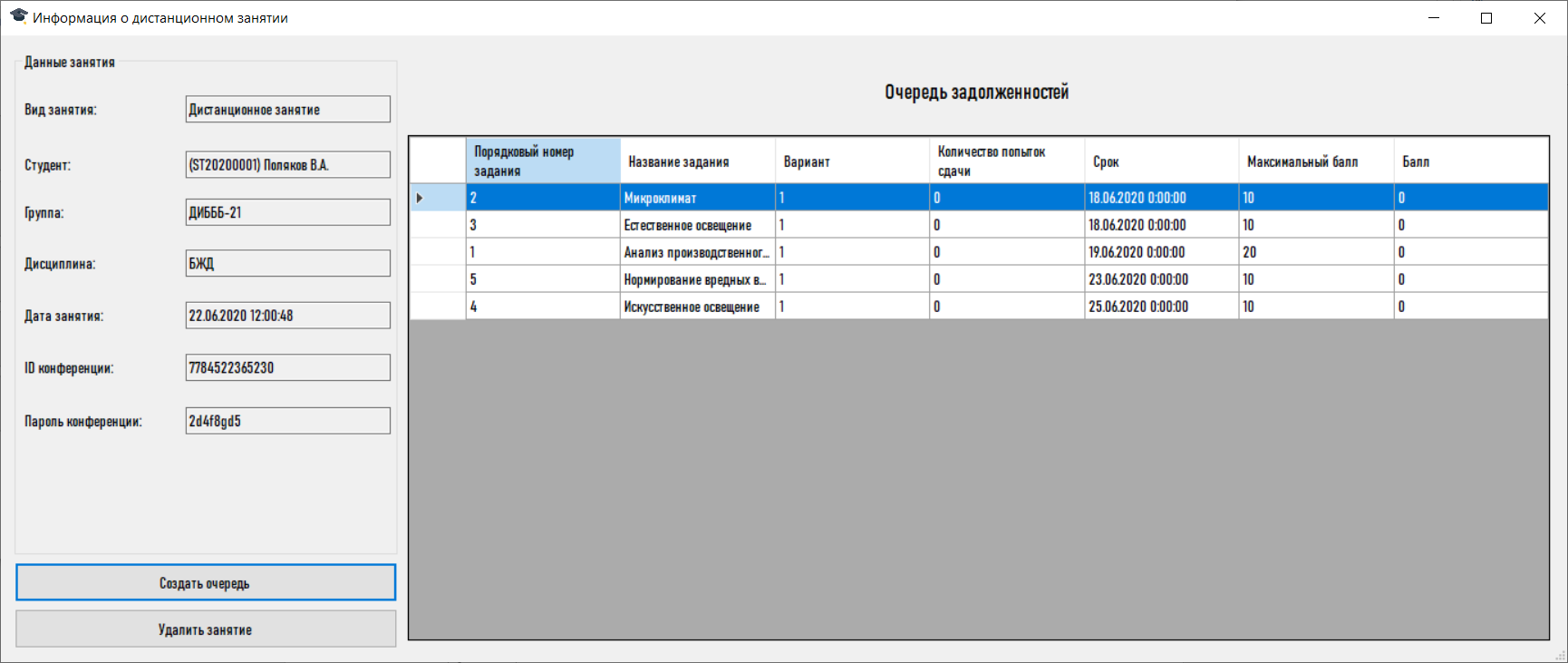


Рисунок 2.27 – Очередь для дистанционного занятия

Для выставления оценки необходимо нажать дважды левой кнопкой мыши или единожды правой кнопкой мыши по студенту и выбрать «Выставить оценку».

**2.8 Сообщения системы**

Чтобы избежать ошибок, которые могут появиться в процессе использования программного продукта вследствие ввода некорректных значений или выполнения других неверных действий пользователя, был реализован механизм обработки исключений. Большинство ошибок, возникающих в ходе работы программного продукта, проявляются в виде сообщений системы. Рассмотрим сообщения системы, с которыми пользователь может встретиться в процессе использования программного продукта.

**2.8.1 Незаполненные поля**

Если пользователь при заполнении данных о студенте, группе, дисциплине, задании или занятии не заполнил какое-либо поле, то появится сообщение об ошибке (рис. 2.28), после чего пользователю предлагается заполнить эти поля снова.

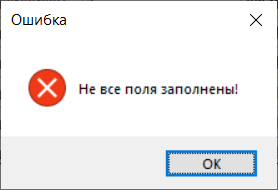


Рисунок 2.28 – Ошибка о незаполненных полях

**2.8.2 Изменение группы у дисциплины**

Если пользователь решил изменить у созданной ранее дисциплины учебную группу, то появится информационное сообщение с подтверждением (рис. 2.29).

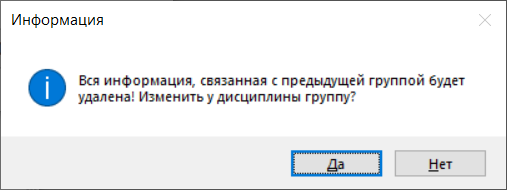


Рисунок 2.29 – Подтверждение об изменении группы у дисциплины

При нажатии на «Да», все задания, занятия и оценки по данной дисциплине будут удалены, в противном случае (при нажатии на «Нет») пользователю снова будет представлено диалоговое окно об изменении дисциплины.

**2.8.3 Существующее название добавляемой группы**

Если пользователь создал группу с названием, которое уже существует, то появится сообщение об ошибке (рис. 2.30), после чего пользователю снова будет представлено диалоговое окно о добавлении группы.

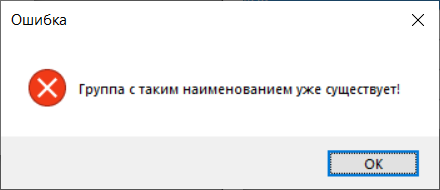


Рисунок 2.30 – Ошибка о существующем названии добавляемой группы

**2.8.4 Изменение группы у студента**

Если пользователь решил изменить у созданного ранее студента учебную группу, то появится информационное сообщение с подтверждением (рис. 2.31).

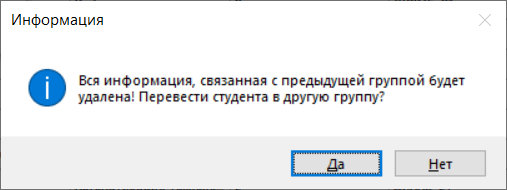


Рисунок 2.31 – Подтверждение об изменении группы у студента

При нажатии на «Да», все оценки и дистанционные занятия данного студента будут удалены, в противном случае (при нажатии на «Нет») пользователю снова будет представлено диалоговое окно об изменении студента.

**2.8.5 Некорректная дата у задания**

Если пользователь заполнил все поля, но установил срок сдачи раньше времени, которое установлено на компьютере (т.е. задним числом), то программа выдаст соответствующую ошибку (рис. 2.32).

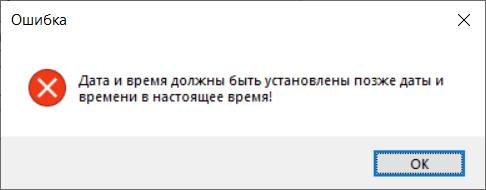


Рисунок 2.32 – Ошибка назначения срока задания 1

Если пользователь ввел срок сдачи позже даты, установленной на компьютере, и разница между соответствующими датами меньше 3-х дней, то программа выдаст сообщение об ошибке (рис. 2.33).

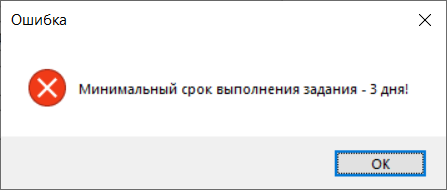


Рисунок 2.33 – Ошибка назначения срока задания 2

**2.8.6 Некорректный максимальный балл у задания**

Если пользователь ввел максимальный балл у задания больше 100, то программа выдаст сообщение об ошибке (рис. 2.34).

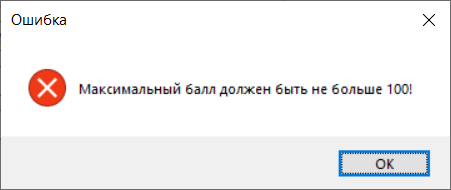


Рисунок 2.34 – Ошибка назначения максимального балла заданию 1

Если пользователь ввел максимальный балл у задания меньше 5-ти, то программа выдаст сообщение об ошибке (рис. 2.35).

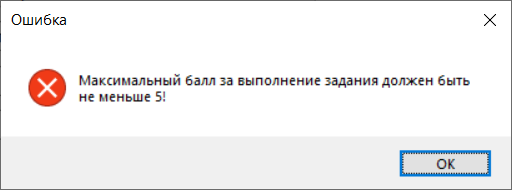


Рисунок 2.35 – Ошибка назначения максимального балла заданию 2

**2.8.7 Существующий порядковый номер у добавляемого задания**

Если пользователь создал задание с порядковым номером, который уже существует, то появится сообщение об ошибке (рис. 2.36), после чего пользователю снова будет представлено диалоговое окно о добавлении задания.

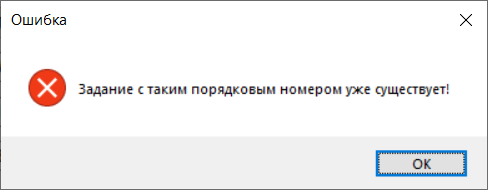


Рисунок 2.36 – Ошибка о существующем порядковом номере добавляемого задания

**2.8.8 Поздняя дата проведения текущего занятия**

Если дата у создаваемого текущего занятия будет позже срока задания, которое планируется отчитывать на этом занятии, то программа выдаст сообщение об ошибке (рис. 2.37).

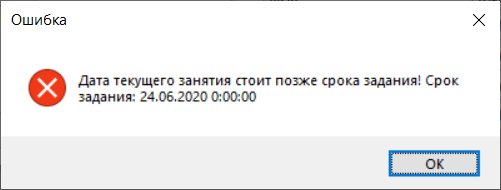


Рисунок 2.37 – Ошибка о поздней дате проведения занятия

**2.8.9 Пустое поле оценки**

Если при выставлении оценки пользователь оставит соответствующее поле пустым, то программа выдаст сообщение об ошибке (рис. 2.38).

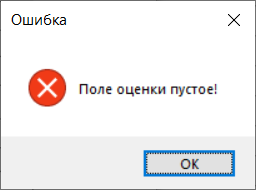


Рисунок 2.38 – Ошибка о незаполненном поле оценки

**2.8.10 Оценка превышает максимальную оценку за задание**

Если выставляемая пользователем оценка превышает максимальную оценку по заданию, то программа выдаст сообщение об ошибке (рис. 2.39).

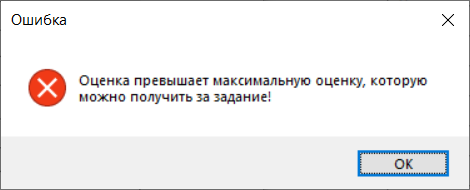


Рисунок 2.39 – Ошибка о максимальной оценке

**2.8.11 Неудовлетворительная оценка у студента**

Если выставляемая оценка составляем меньше 60-ти процентов от максимальной то программа выдаст информационное сообщение с подтверждением (рис. 2.40).

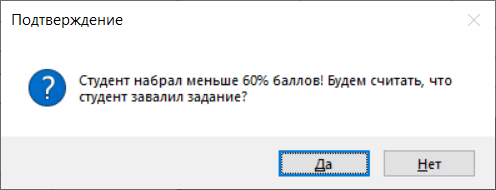


Рисунок 2.40 – Сообщение об неудовлетворительной оценке

При нажатии на «Да» количество попыток сдать задание у студента увеличится на 1, а в поле оценки будет стоять тот балл, который ввел пользователь, в противном случае (при нажатии на «Нет») пользователю снова будет представлено диалоговое окно о выставлении оценки.