МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
"Южно-Уральский государственный университет (национальный исследовательский университет)"
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

ОТЧЕТ

по учебной практике

бакалавра направления 09.03.04 "Программная инженерия"

Выполнил:_____
студент группы КЭ-204
Емельянов Г.С.

Проверил:_____
Преподаватель кафедры СП
Быков Н.С.
Дата:_____, Оценка:_____

Челябинск-2023

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
"Южно-Уральский государственный университет (национальный исследовательский университет)"
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ

Зав. кафедрой
системного программирования

_____ Л.Б. Соколинский

ЗАДАНИЕ
по учебной практике

1. **Цель работы**

Разработать GUI-приложение, работающее с входной информацией, вводимой пользователем с помощью управляемых элементов формы, либо из текстового файла.

2. **Исходные данные к работе**

1. База данных экзаменов/зачетов, содержащая поля: дисциплина, перечень тем, семестр изучения, оценка (0-100 баллов), дата получения оценки (более 60 баллов, если положительная оценка не получена, дата не заполняется).
2. Данные хранятся в файле в формате txt.
3. Данные, которые хранятся во входном файле: disciplineName, themes, semesterNumber, rating, dateOfExam.

3. **Перечень подлежащих разработке вопросов**

1. Определение структуры приложения (по модулям), структур данных, используемых для хранения основной пользовательской информации.
2. Дизайн оконного интерфейса, анализ структуры входных данных и их защита от некорректного ввода информации.
3. Разработка основного функционала приложения: основных форм и механизмов получения информации из их компонентов и их файлов; основного алгоритма функционирования приложения; тестирование приложения.
4. Подготовка руководства пользователя и документации для программиста.

4. **Сроки**

Дата выдачи задания: "26" июня 2023 г.
Срок сдачи законченной работы: "23" июля 2023 г.

Руководитель:

Преподаватель кафедры СП	_____	Быков Н.С.
должность, ученая степень руководителя	подпись	ФИО

Задание принял к исполнению:

_____	Емельянов Г.С.
подпись	ФИО студент

ОГЛАВЛЕНИЕ

1. ПОСТАНОВКА ЗАДАЧИ

Требуется написать программу для студента, который ведет электронную зачетную книжку (ЭЗК) по всем изучаемым предметам. Формат записи ЭЗК: дисциплина, перечень изучаемых тем, семестр изучения, оценка, дата получения положительной оценки. Программа осуществляет добавление и редактирование имеющихся записей, их удаление, поиск по 3 разным ключам.

Записи ЭЗК будут иметь следующие типы (табл.1).

Табл. 1. Переменные, используемые для хранения записей ЭЗК

Переменная	Тип переменной	Содержательный смысл
disciplineName	string	Название дисциплины
themes	string	Перечень изучаемых тем
semesterNumber	int	Семестр изучения
rating	float	Оценка
dateOfExam	string	Дата проставления оценки
id_	int	Идентификационный номер

Реализация решения данной задачи осуществляется с помощью двунаправленного связанного списка, класса List. Каждый его элемент будет включать конкретную дисциплину. Двусвязный список хранит информацию о дисциплинах и находится в отдельном классе (Листинг 2). Описание конкретной дисциплины осуществляется в отдельно существующем классе (Листинг 1).

Листинг 1. Объявление класса Discipline в Discipline.h

```
ref class Discipline
{
public:
	System::String^ disciplineName_;
	System::String^ themes_;
	int semestrNumber_;
	float rating_;
	System::String^ dateOfExam_;
	int id_;
	Discipline(int id, System::String^ disciplineName, System::String^
themes, int semestrNumber, float rating, System::String^ dateOfExam);
	void setId(int id);
	};
```

Листинг 2. Объявление класса DisciplineManager в DisciplineManager.h, реализация методов класса Discipline

```
ref class DisciplineManager
{
public:
	List<Discipline^>^ disciplines;
	DisciplineManager();
	int id;
	int fromFile(std::string filename);
	int toFile(std::string filename);
	int addDiscipline(Discipline^ discipline);
	void deleteDiscipline(int id);
	void editDiscipline(Discipline^ discipline, const int id);
	List<Discipline^>^ findByName(System::String^ name);
	List<Discipline^>^ findBySemestr(int semNum);
	List<Discipline^>^ findByRating(float rating);
};
```

Разрабатываемое приложение состоит из пяти оконных форм:

1. Главное окно программы (кнопки для осуществления всех основных функций: добавления, удаления, изменения и поиска данных).
2. Окно для добавления дисциплины (добавление новой дисциплины, заполнение информации о ней).
3. Окно для удаления дисциплины (удаление уже существующей дисциплины).
4. Окно для изменения дисциплины (изменение данных уже имеющихся дисциплин).
5. Окно для поиска необходимых данных (с помощью определенных критериев отбора, ключей).

Каждой из разработанных оконных форм соответствует пара файлов (*.h и *.cpp), приведенные в таблице 2.

Табл. 2. Модули создаваемого проекта

Имя файла	**Описание информации, содержащейся в нем**	**Функциональное назначение**	**Файлы проекта, подключенные к текущему файлу посредством директивы #include**
Discipline.cpp	Реализация конструкторов класса Discipline	Работа с экземпляром класса Discipline	Discipline.h

Имя файла	Описание информации, содержащейся в нем	Функциональное назначение	Файлы проекта, подключенные к текущему файлу посредством директивы #include
Discipline.h	Описание класса Discipline и конструкторов	Объявление класса Discipline	Нет
DisciplineManager.h	Описание класса DisciplineManager и его методов	Объявление класса DisciplineManager	Discipline.h
DisciplineManager.cpp	Реализация класса DisciplineManager и его методов	Работа с данными списка	DisciplineManager.h
MainWindow.cpp	Стартовое окно	Отображение стартового окна программы, меню перехода на другие окна	MainWindow.h
MainWindow.h	Описание интерфейса главного окна	Реализация интерфейса главного окна	AddDisciplineWindow.h RemoveDisciplineWindow.h EditDisciplineWindow.h SearchDisciplineWindow.h DisciplineManager.h
AddDisciplineWindow.cpp	Окно для добавления новой дисциплины	Добавление данных дисциплины в зачетную книжку	AddDisciplineWindow.h
AddDisciplineWindow.h	Описание интерфейса окна добавления дисциплины	Реализация интерфейса окна для добавления новой дисциплины	DisciplineManager.h
RemoveDisciplineWindow.cpp	Окно для удаления существующих дисциплин	Удаление данных о дисциплине	RemoveDisciplineWindow.h
RemoveDisciplineWindow.h	Описание интерфейса окна удаления дисциплины	Реализация интерфейса окна для удаления существующей дисциплины	DisciplineManager.h
EditDisciplineWindow.cpp	Окно для изменения конкретной дисциплины	Изменение данных дисциплины в зачетной книжке	EditDisciplineWindow.h
EditDisciplineWindow.h	Описание интерфейса окна изменения данных дисциплины	Реализация интерфейса окна для	DisciplineManager.h

Имя файла	Описание информации, содержащейся в нем	Функциональное назначение	Файлы проекта, подключенные к текущему файлу посредством директивы #include
		изменения дисциплины	
SearchDisciplineWindow.cpp	Окно для поиска информации по конкретным данным	Выбор параметров для поиска информации по конкретной дисциплине	SearchDisciplineWindow.h
SearchDisciplineWindow.h	Описание интерфейса окна поиска данных в зачетной книжке	Реализация интерфейса окна для поиска данных в зачетной книжке	DisciplineManager.h

Используя информацию из табл.2 получим структурную схему разрабатываемого приложения (Рис.1).

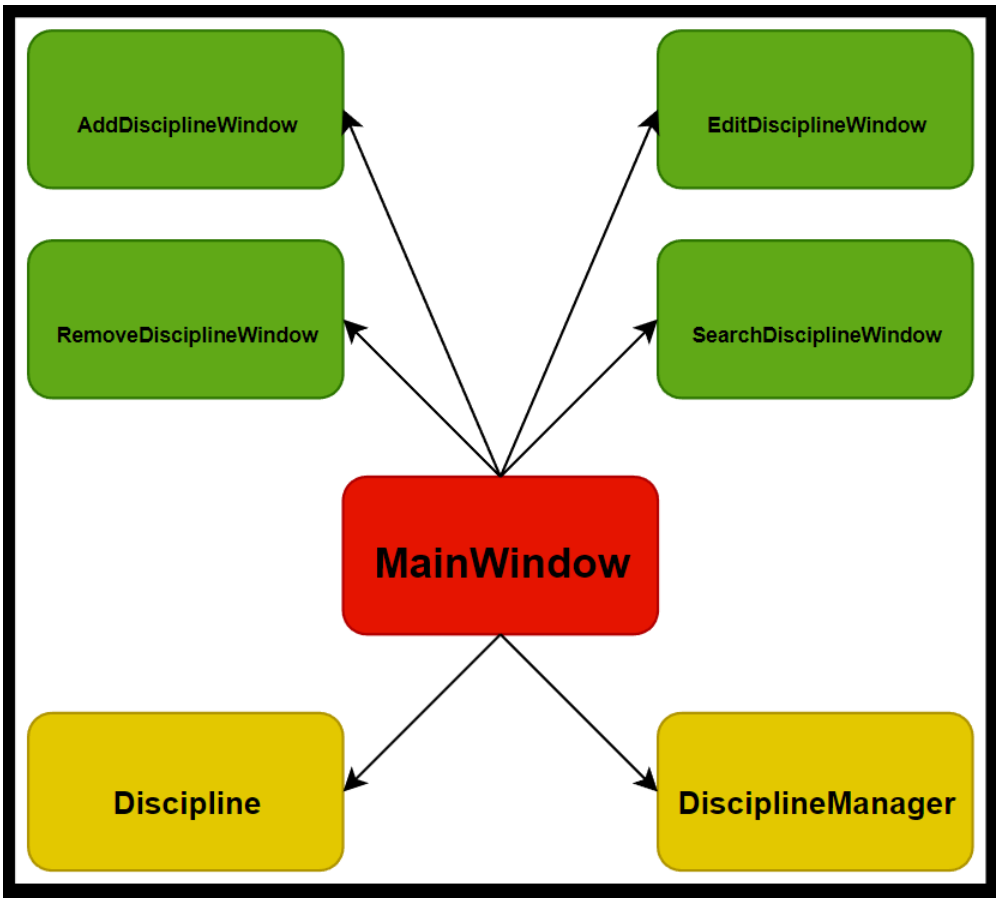

Рисунок 1. Структурная схема приложения

2. ДИЗАЙН ОКОННОГО ИНТЕРФЕЙСА

Ниже приводится описание всех оконных форм, используемых для функционирования приложения.

Разрабатываемое приложение состоит из пяти оконных форм:

1. Главное окно программы.
2. Окно для добавления дисциплины.
3. Окно для удаления дисциплины.
4. Окно для изменения дисциплины.
5. Окно для поиска необходимых данных.

Реализация электронной зачетной книги для работы с данными студента осуществляется через следующие формы (рис.2-6).

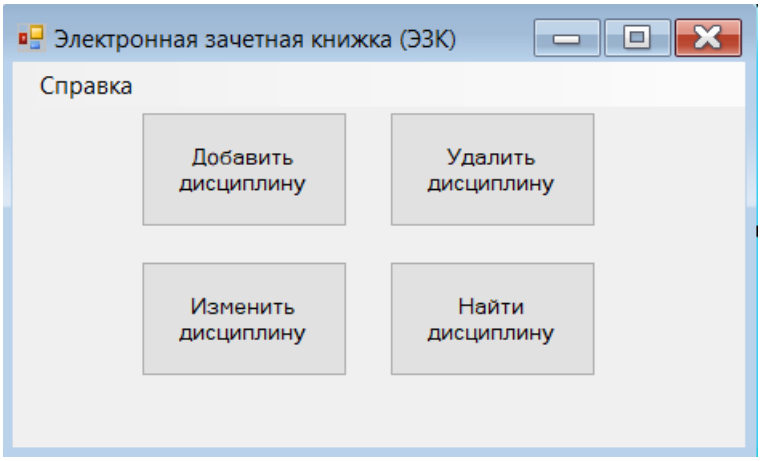

Рисунок 2. Оформление формы основного окна приложения

Размещенные на формах компоненты и перечень методов и событий, которые необходимо реализовать приведен в табл.3-7.

Табл. 3. Компоненты основного окна приложения

Имя компоненты	**Тип**	**Ограничения для ввода информации**	**Реализованные события**	**Функциональное назначение**
AddDiscButton	button	Инициализация открытия окна добавления новой дисциплины	Click	При нажатии кнопки осуществляется переход на форму добавления дисциплины

Имя компоненты	Тип	Ограничения для ввода информации	Реализованные события	Функциональное назначение
RemoveDiscButton	button	Инициализация открытия окна удаления дисциплины	Click	При нажатии кнопки осуществляется переход на форму удаления дисциплины
EditDiscButton	button	Инициализация открытия окна изменения данных дисциплины	Click	При нажатии на кнопку осуществляется переход на форму изменения дисциплины
SearchDiscButton	button	Инициализация открытия окна поиска данных о дисциплинах	Click	При нажатии на кнопку осуществляется переход на форму поиска данных по дисциплине
справкаToolStripMenuItem	ToolStripMenu	Инициализация открытия меню	Click	При нажатии кнопки осуществляется “выпад” меню с набором: -О программе (версия) -Об авторе (ФИО) -Руководство пользователя

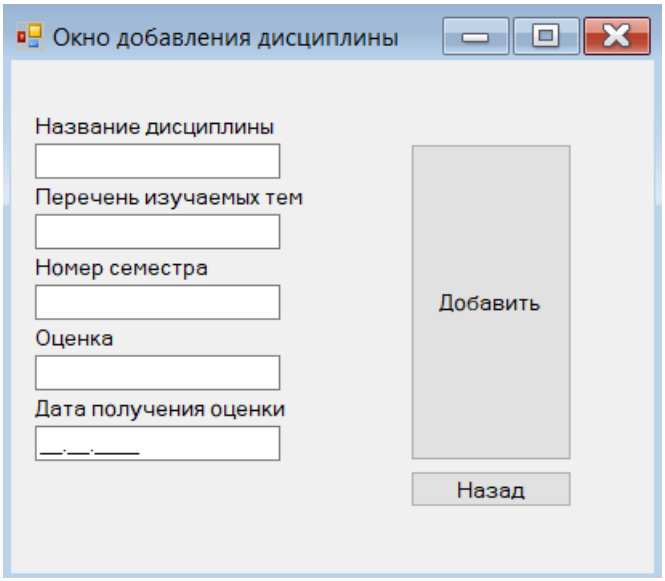

Рисунок 3. Оформление формы окна для добавления дисциплины

Табл. 4. Компоненты окна для добавления дисциплины

Имя компоненты	**Тип**	**Ограничения для ввода информации**	**Реализованные события**	**Функциональное назначение**
AddDisciplineNameTb	textBox	Ввод с клавитатуры только букв	KeyPress (для проверки вводимой информации)	Пользователь задает название дисциплины
AddThemesTb	textBox	Ввод с клавитатуры только букв	KeyPress (для проверки вводимой информации)	Пользователь задает изучаемые темы дисциплины
AddSemesterNumberTb	textBox	Ввод с клавиатуры только цифр	KeyPress (для проверки вводимой информации)	Пользователь задает номер семестра
AddRatingTb	textBox	Ввод с клавиатуры только цифр	KeyPress (для проверки вводимой информации)	Пользователь задает оценку (баллы) за дисциплину
AddDateOfExamTb	maskedTextBox	Ввод данных по маске даты 00/00/0000	Нет	Пользователь задает дату выставления оценки согласно маске
BackButton	button	Инициализация открытия основного окна приложения	Click	При нажатии на кнопку осуществляется переход на форму

Имя компоненты	Тип	Ограничения для ввода информации	Реализованные события	Функциональное назначение
				основного окна приложения
AddButton	button	Инициализация добавления новой дисциплины	Click	При нажатии на кнопку осуществляется добавление новой дисциплины

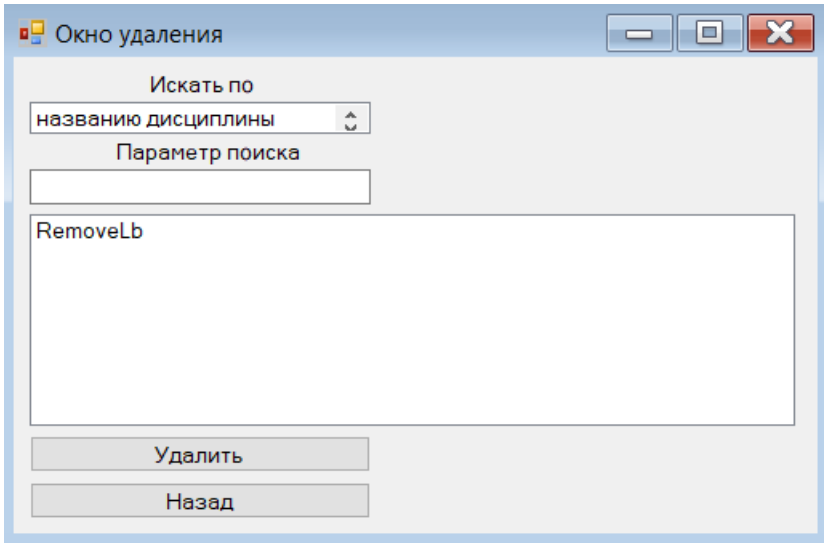

Рисунок 4. Оформление формы окна для удаления дисциплины

Табл. 5. Компоненты окна для удаления дисциплины

Имя компоненты	Тип	Ограничения для ввода информации	Реализованные события	Функциональное назначение
SearchRemByLb	ListBox	Инициализация выбора типа данных для поиска	SelectedIndexChanged	Выбор типа данных по которым будет осуществлен поиск удаляемой дисциплины
SearchRemTb	textBox	Нет	TextChanged	При вводе параметра поиска в RemoveLB появляется перечень подходящих под параметр данных

Имя компоненты	Тип	Ограничения для ввода информации	Реализованные события	Функциональное назначение
RemoveLb	ListBox	Инициализация выбора удаления дисциплины	SelectedIndexChanged	Выбор удаляемой дисциплины
BackButton	button	Инициализация открытия основного окна приложения	Click	При нажатии на кнопку осуществляется переход на форму основного окна приложения
RemoveButton	button	Инициализация удаления дисциплины	Click	При нажатии на кнопку осуществляется удаление выбранной дисциплины

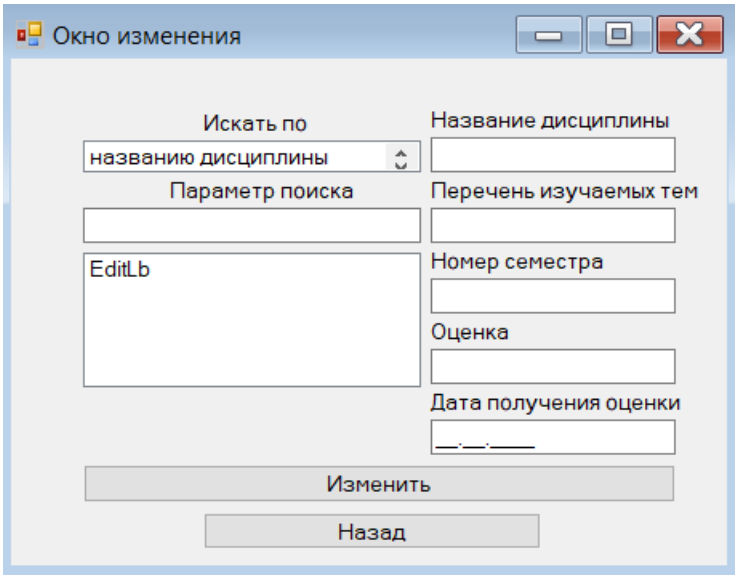

Рисунок 5. Оформление формы окна для изменения дисциплины

Табл. 6. Компоненты окна для изменения дисциплины

Имя компоненты	Тип	Ограничения для ввода информации	Реализованные события	Функциональное назначение
SearchEditByLb	ListBox	Инициализация выбора типа данных для поиска	SelectedIndexChanged	Выбор типа данных по которым будет осуществлен

Имя компоненты	Тип	Ограничения для ввода информации	Реализованные события	Функциональное назначение
				поиск изменяемой дисциплины
SearchEditTb	textBox	Нет	TextChanged	При вводе параметра поиска в EditLB появляется перечень подходящих под параметр данных
EditLb	ListBox	Инициализация выбора изменения дисциплины	SelectedIndexChanged	Выбор изменяемой дисциплины и автоматическое заполнение данных выбранной дисциплины для дальнейшего редактирования.
EditDisciplineNameTb	textBox	Ввод с клавитатуры только букв	KeyPress (для проверки вводимой информации)	Пользователь задает название дисциплины
EditThemesTb	textBox	Ввод с клавитатуры только букв	KeyPress (для проверки вводимой информации)	Пользователь задает изучаемые темы дисциплины
EditSemesterNumberTb	textBox	Ввод с клавиатуры только цифр	KeyPress (для проверки вводимой информации)	Пользователь задает номер семестра
EditRatingTb	textBox	Ввод с клавиатуры только цифр	KeyPress (для проверки вводимой информации)	Пользователь задает оценку (баллы) за дисциплину
EditDateOfExamTb	maskedTextBox	Ввод данных по маске даты 00/00/0000	Нет	Пользователь задает дату выставления оценки согласно маске
BackButton	button	Инициализация открытия основного окна приложения	Click	При нажатии на кнопку осуществляется переход на форму основного окна приложения
EditButton	button	Инициализация изменения данных о дисциплине	Click	При нажатии на кнопку осуществляется изменение

Имя компоненты	Тип	Ограничения для ввода информации	Реализованные события	Функциональное назначение
				выбранной дисциплины

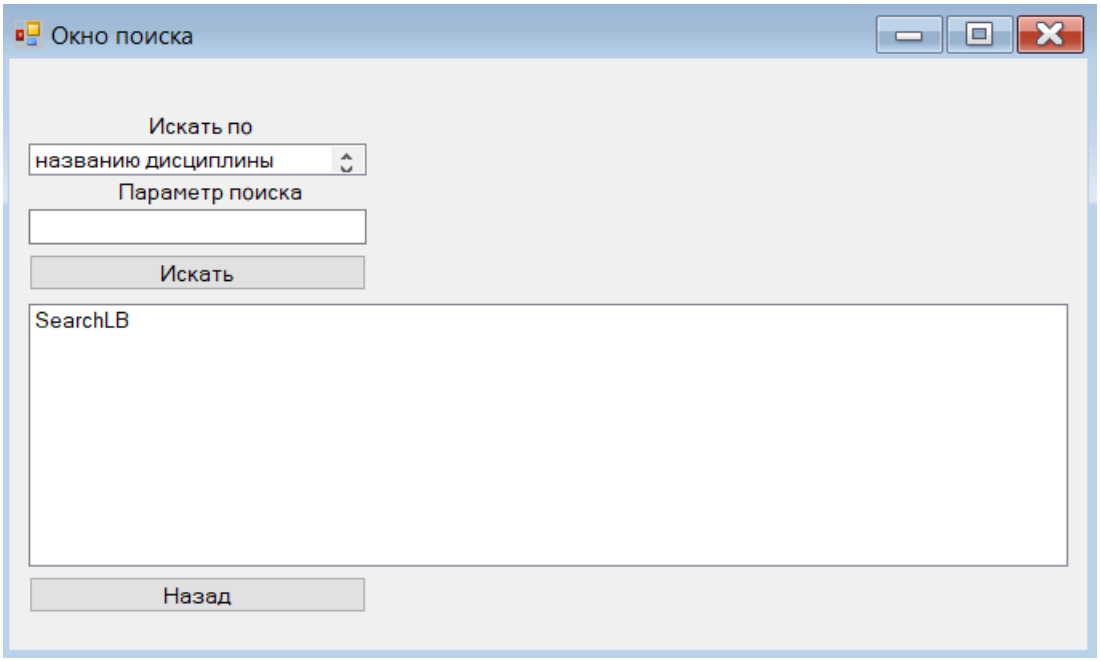

Рисунок 6. Оформление формы окна для поиска данных о дисциплинах

Табл. 7. Компоненты окна для поиска данных о дисциплинах

Имя компоненты	Тип	Ограничения для ввода информации	Реализованные события	Функциональное назначение
SearchByLb	ListBox	Инициализация выбора типа данных для поиска	SelectedIndexChanged	Выбор типа данных по которым будет осуществлен поиск изменяемой дисциплины
SearchTb	textBox	Нет	TextChanged	Ввод параметра для поиска данных
SearchLb	ListBox	Инициализация выбора изменения дисциплины. Ввод запрещен	Нет	Просмотр найденных дисциплин
SearchButton	button	Инициализация поиска данных дисциплин	Click	При нажатии на кнопку осуществляется поиск по данным

Имя компоненты	Тип	Ограничения для ввода информации	Реализованные события	Функциональное назначение
				SearchTB дисциплин и их вывод в SearchLB
BackButton	button	Инициализация открытия основного окна приложения	Click	При нажатии на кнопку осуществляется переход на форму основного окна приложения

Алгоритмы и программная реализация приведенных в табл. 3-7 событий и методов приведены в следующем разделе.

3. РАЗРАБОТКА ФУНКЦИОНАЛА ОСНОВНЫХ ФОРМ И МЕХАНИЗМОВ ПОЛУЧЕНИЯ ИНФОРМАЦИИ

3.1. Стартовое окно (MainWindow.h)

Главное меню используется для перехода в другие формы. При запуске открывается стартовое окно с кнопками, с помощью которых можно перейти на другие формы. На листинге 3 приведен метод для открытия окна с добавлением записи, методы для остальных окон в программе описаны аналогично.

Листинг 3. Метод открытия окна добавления.

```
private: System::Void AddButton_Click(System::Object^ sender,
System::EventArgs^ e)
	{
		AddWindow^ addwin = gcnew AddWindow();
		(addwin->discman).disciplines = discmanMain.disciplines;
		(addwin->discman).id = discmanMain.id;
		addwin->Owner = this;
		addwin->Show();
		this->Hide();
	}
```

3.2. Окно добавления (AddWindow.h)

В данном окне происходит добавления новой записи дисциплины. После заполнения необходимых полей и нажатия кнопки «Добавить», данные сохраняются в файл, на листинге 4 приведен этот метод. Метод сохранения дисциплины в файл CSV реализован в DisciplineManager.cpp.

Листинг 4. Метод добавления новой записи.

```
private: System::Void AddButton_Click(System::Object^ sender,
System::EventArgs^ e)
	{

		if (errorMessage()->Length > 0)
		{
			MessageBox::Show(errorMessage(), "Ошибка заполнения
полей!");
		}
		else
		{
			String^ name = AddDisciplineNameTb->Text;
			String^ themes = AddThemesTb->Text;
			int semesterNumber = Int32::Parse(AddSemesterNumberTb-
>Text);
			float rating = float::Parse(AddRatingTb->Text);
			String^ dateOfExam = AddDateOfExamTb->Text;
			Discipline^ newDiscipline = gcnew Discipline(0, name,
themes, semesterNumber, rating, dateOfExam);
```

```
            MessageBox::Show("Запись успешно добавлена!");

            AddDisciplineNameTb->Text = "";
            AddThemesTb->Text = "";
            AddSemesterNumberTb->Text = "";
            AddRatingTb->Text = "";
            AddDateOfExamTb->Text = "";

            discman.addDiscipline(newDiscipline);
            discman.toFile("data.txt");
        }
    }
```

Схема, изображенная на рисунке 7, показывает систему приведения типа переменных к тому типу, который используется в базе данных.

Рисунок 7. Схема приведения типа переменных.

3.3. Окно поиска (SearchWindow.h)

В данном окне пользователь может найти запись по выбранному параметру. Пользователь выбирает в SearchByLb по какому параметру искать запись (названию дисциплины, семестру изучения, дате сдачи) и вводит ключевые слова в SearchTb. По нажатии кнопки «Искать» будет выполнен поиск записей по выше описанным условиям и все возможные совпадения будут выведены в SearchLB, реализация метода описана в листинге 5.

Листинг 7. Поиск информации по названию и её вывод.

```
private: System::Void SearchButton_Click(System::Object^ sender,
System::EventArgs^ e)
    {
        SearchLB->Items->Clear();

        if (SearchByLb->Text == "названию дисциплины")
        {
            System::String^ const name = SearchTb->Text;
            if (name->Length > 0)
            {
```

```
                        List<Discipline^>^ finded =
discman.findByName(name);
                        for (int i = 0; i < finded->Count; i++)
                        {
                              System::String^ out = finded[i]->id_ + ": " +
finded[i]->disciplineName_ + " - " + finded[i]->themes_ + " - " +
finded[i]->semestrNumber_ + " - " + finded[i]->rating_ + " - " + finded[i]-
>dateOfExam_;
                              SearchLB->Items->Add(out);
                        }

                        if (SearchLB->Items->Count == 0)
                              MessageBox::Show("Таких дисциплин нет в
файле!");
                  }
                  else
                        MessageBox::Show("Дисциплина должна иметь
название!");

            else
            {
                  MessageBox::Show("Выберете параметр поиска!");
            }
      }
```

3.4. Окно удаления (RemoveWindow.h)

В данном окне происходит удаления записи. Происходит поиск по алгоритму, аналогичному в окне поиска, только здесь не требуется нажимать на кнопку для поиска. Вместо этого здесь реализован метод TextChanged в SearchRemTb. При любом изменении список подходящих дисциплин будет обновляться в RemoveLb. По нажатии кнопки «Удалить» происходит удаление дисциплины, реализация этого метода описана в листиниге 8.

Листинг 8. Удаление дисциплины из файла.

```
private: System::Void RemoveButton_Click(System::Object^ sender,
System::EventArgs^ e)
      {
            if (RemoveLb->SelectedItems->Count > 0)
            {
                  String^ str = RemoveLb->SelectedItem->ToString()-
>Split(':')[0];
                  int delId = 0;
                  delId = Int32::Parse(str);
                  discman.deleteDiscipline(delId);
                  discman.toFile("data.txt");
                  MessageBox::Show("Дисциплина успешно удалена!");
                  Owner->Show();
                  this->Close();
            }
            else
            {
                  MessageBox::Show("Дисциплина не выбрана!");
```

```
        }
    }
```

3.5. Окно изменения (EditWindow.h)

В данном окне пользователь может изменить данные записи. Сначала пользователь ищет нужную запись для редактирования, по алгоритму аналогичному в окне удаления. Далее выбирает нужную ему запись из EditLb. Здесь реализован метод SelectedIndexChanged и при каждом новом выборе записи будут автоматически заполняться поля с данными этой записи, которые пользователь может изменять на свое усмотрение. При нажатии кнопки «Изменить» данные будут изменены, если не нажимать эту кнопку, то соответственно изменения не будут сохранены.

Листинг 9. Выбор дисциплины, заполнение полей.

```
private: System::Void EditLb_SelectedIndexChanged(System::Object^ sender,
System::EventArgs^ e)
    {
        String^ str = EditLb->SelectedItem->ToString();
        editId = Int32::Parse(str->Split(':')[0]);
        str = str->Split(':')[1];
        String^ editName = str->Split('-')[0];
        String^ editThem = str->Split('-')[1];
        String^ editSem = str->Split('-')[2];
        String^ editRat = str->Split('-')[3];
        String^ editDate = str->Split('-')[4];

        EditDisciplineNameTb->Text = editName;
        EditThemesTb->Text = editThem;
        EditSemesterNumberTb->Text = editSem;
        EditRatingTb->Text = editRat;
        EditDateOfExamTb->Text = editDate;
    }
```

4. РАЗРАБОТКА ОСНОВНОГО МЕХАНИЗМА ФУНКЦИОНИРОВАНИЯ ПРИЛОЖЕНИЯ

4.1 Чтение информации из файла и запись в него

Открываем файл для чтения через ifstrem. Перед добавлением дисциплин из файла в список очищаем его, затем проходим по каждой строке файла, парсим ее, создаем для каждой экземпляры Discipline и добавляем объект дисциплины в двунаправленный список. Вышеописанный метод приведен в листинге 10. Для записи и чтения файла используется библиотека fstream. Этот метод используется единожды при запуске приложения, далее вся работа происходит над списком.

Листинг 10. Метод для считывания данных из файла.

```
int DisciplineManager::fromFile(std::string filename)
{
	std::ifstream reader;

	reader.open(filename);
	if (!reader.is_open())
	{
		std::ofstream { "data.txt" };
		return -1;
	}
	else
	{
		std::string str;
		id = 0;
		while (std::getline(reader, str))
		{
			int x = 0;
			//str = "";
			id += 1;

			System::String^ disName;
			System::String^ disThem;
			System::String^ date;
			int semNum = 0;
			float rating = 0;

			x = str.find_first_of("; ") + 1;
			std::string sub = str.substr(0, x - 1);
			disName = gcnew System::String(sub.c_str());

			str.erase(0, x+1);

			x = str.find_first_of(";")+1;
			sub = str.substr(0, x - 1);
			disThem = gcnew System::String(sub.c_str());
			str.erase(0, x+1);

			x = str.find_first_of(";")+1;
			sub = str.substr(0, x - 1);
```

```
                  semNum = std::stoi(sub);
                  str.erase(0, x+1);

                  x = str.find_first_of(";") + 1;
                  sub = str.substr(0, x - 1);
                  rating = std::stof(sub);
                  str.erase(0, x+1);

                  date = gcnew System::String(str.c_str());

                  Discipline^ nd = gcnew Discipline(id, disName, disThem,
semNum, rating, date);
                  disciplines->Add(nd);
            }
            reader.close();
            return 0;
      }
      }
```

Для записи данных проходим по всему списку. Каждый объект хранится в отдельной строке, а его атрибуты разделяются «;». Этот метод приведен в листинге 11. Данные метод также используется единственный раз, перед закрытием приложения.

Листинг 11. Метод для записи данных в файл.

```
int DisciplineManager::toFile(std::string filename)
{
      std::ofstream reader;
      reader.open(filename);
      if (!reader.is_open())
      {
            return -1;
      }
      else
      {
            for (int i = 0; i < disciplines->Count; i++)
            {
                  msclr::interop::marshal_context context;
                  std::string s;
                  s += context.marshal_as<std::string>(disciplines[i]-
>disciplineName_);
                  s += "; " +
context.marshal_as<std::string>(disciplines[i]->themes_);
                  s += "; " +
context.marshal_as<std::string>(disciplines[i]->semestrNumber_.ToString());
                  s += "; " +
context.marshal_as<std::string>(disciplines[i]->rating_.ToString());
                  s += "; " +
context.marshal_as<std::string>(disciplines[i]->dateOfExam_);
                  reader << s << std::endl;
            }
      }
      reader.close();
      return 0;
      }
```

4.2 Добавление дисциплины

Для хранения дисциплин использовалась структура данных list, представляющая собой двусвязный список, наполненный экземплярами класса Discipline. В конец списка добавляется переданный в метод экземпляр класса с помощью стандартного метода push_back. Метод добавления описан в листинге 12.

Листинг 12. Метод добавления дисциплины в список.

```
int DisciplineManager::addDiscipline(Discipline^ discipline)
{
	id += 1;
	discipline->setId(id);
	disciplines->Add(discipline);

	return 0;
	}
```

4.3 Удаление дисциплины

Для удаления дисциплины пользователь сначала ее выбирает из предложенного списка. В метод поступает уникальный id того экземпляра, который требуется его удалить. Далее в цикле ищется этот id и удаляется из списка. Данный метод представлен в листинге 13.

Листинг 13. Метод удаления дисциплины из списка.

```
void DisciplineManager::deleteDiscipline(int id)
{
	for (int i = 0; i < disciplines->Count; i++)
	{
		if (disciplines[i]->id_ == id)
		{
			disciplines->RemoveAt(i);
			break;
		}
	}
	}
```

4.4 Редактирование дисциплины

При редактировании информации в списке используются другие методы работы с базой данных: добавления и удаления дисциплин. Сначала удаляем информацию о выбранной дисциплине из списка, затем обновляем информацию об этой же дисциплине, путем добавления новой – с тем же id. В метод также поступают оба параметра, такие как обновленная дисциплина и ее id. Метод редактирования можно увидеть в листинге 14.

Листинг 14. Метод редактирования дисциплины.

```
void DisciplineManager::editDiscipline( Discipline^ discipline, const int
id)
{
	deleteDiscipline(id);
	disciplines->Add(discipline);
	}
```

4.5 Поиск данных

Методы поиска реализованы независимо друг от друга, каждый включает в себя определенный критерий поиска: название дисциплины, оценку и семестр. В листинге 15 реализован метод поиска по названию, листинге 16 описан метод поиска по семестру, листинге 17 – по оценке. Основной принцип работы методов: перебор данных в списке для поиска определенного критерия, при нахождении – добавление найденной дисциплины в возвращаемый список.

Листинг 14. Метод для поиска информации по названию дисциплины

```
List<Discipline^>^ DisciplineManager::findByName(System::String^ name)
{
	List<Discipline^>^ finded = gcnew List<Discipline^>;
	for (int i = 0; i < disciplines->Count; i++)
	{
		if (disciplines[i]->disciplineName_ == name)
			finded->Add(disciplines[i]);
	}
	return finded;
}
```

Листинг 15. Метод для поиска информации по семестру

```
List<Discipline^>^ DisciplineManager::findBySemestr(int semNum)
{
	List<Discipline^>^ finded = gcnew List<Discipline^>;
	for (int i = 0; i < disciplines->Count; i++)
	{
		if (disciplines[i]->semestrNumber_ == semNum)
			finded->Add(disciplines[i]);
	}
	return finded;
}
```

Листинг 16. Метод для поиска информации по оценке

```
List<Discipline^>^ DisciplineManager::findByRating(float rating)
{
	List<Discipline^>^ finded = gcnew List<Discipline^>;
	for (int i = 0; i < disciplines->Count; i++)
	{
		if (disciplines[i]->rating_ == rating)
			finded->Add(disciplines[i]);
	}
	return finded;
}
```

5. ТЕСТИРОВАНИЕ

Данный раздел разбит на следующие пункты: автономное и комплексное тестирование.

5.1. Автономное тестирование

В данном разделе приведены протоколы тестирования всех основных механизмов, приведенных в главе 4: чтение и сохранение файла, редактирование, поиск по разным ключам, удаление и добавление элемента. Результаты тестирования отображены в таблицах 8-12.

Табл.8. Тестирование операции чтения и сохранения данных.

Входные данные	**Выходные данные**	**Верно ли?**
data.txt	Чтение данных из файла	Да
data.txt	Сохранение данных в файл	Да

Табл. 9. Тестирование поиска данных по ключам

Входные данные	**Выходные данные**	**Верно ли?**
Ввод пользователем в поля: String^ SearchTb корректных данных, передаваемых в функцию	Данные успешно введены	Да
Ввод пользователем данных в поля: String^ SearchTb приводящих к ошибке ввода (данные: отрицательные числа, , пустая строка)	Вывод всплывающего окно, сообщающего об ошибке	Да

Табл. 10. Тестирование редактирования данных.

Входные данные	**Выходные данные**	**Верно ли?**
Ввод пользователем в поля: String^ EditDisciplineNameTb,String^ EditThemesTb , Int EditSemesterNumberTb, float EditRatingTb, String^ EditDateOfExamTb	Данные успешно введены	Да

Входные данные	Выходные данные	Верно ли?
корректных данных, предаваемых в функцию		
Ввод пользователем данных в поля: String^ AddDateOfExamTb, Int AddSemesterNumberTb, приводящих к ошибке ввода (данные: отрицательные числа, буквы, пустая строка, ввод не по маске)	Вывод всплывающего окно, сообщающего об ошибке ввода	Да

Табл. 11. Тестирование добавления данных.

Входные данные	Выходные данные	Верно ли?
Ввод пользователем в поля: String^ AddDisciplineNameTb,String^ AddThemesTb , Int AddSemesterNumberTb, float AddRatingTb, String^ AddDateOfExamTb корректных данных, предаваемых в функцию	Данные успешно введены	Да
Ввод пользователем данных в поля: Int AddSemesterNumberTb (данные: отрицательные числа, пустая строка, буквы), float AddRatingTb (данные: отрицательные числа, пустая строка, буквы), приводящих к ошибке ввода.	Вывод всплывающего окно, сообщающего об ошибке ввода	Да

Табл. 12. Тестирование удаления данных.

Входные данные	Выходные данные	Верно ли?
То же, что и в табл.10, а также: Пользователь выбрал дисциплину для удаления.	Данные успешно введены	Да
То же, что и в табл.10, а также: Пользователь НЕ выбрал дисциплину для удаления.	Вывод всплывающего окно, сообщающего об ошибке	Да

5.2. Комплексное тестирование

В данном разделе приведены протоколы тестирования приложения. В таблицах 13 - 17 описаны протоколы тестирования для всех форм, описанных в главе 3 отчета.

Табл.13. Тестирование основной формы приложения.

№ п/п	Описание ситуации	Входные данные	Выходные данные	Тест пройден?
1.	Пользователь хочет найти дисциплину и открывает соответствующую форму путем нажатия кнопки «Найти дисциплину»	нет	Открытие окна с поиском дисциплины	Да
2.	Пользователь хочет редактировать дисциплину и нажимает на соответствующую кнопку	нет	Открытие окна с редактированием дисциплины. Блокировка основного окна	да
3.	Пользователь хочет добавить новую дисциплину и нажимает на соответствующую кнопку	нет	Открытие окна с добавлением дисциплины. Блокировка основного окна	да
4.	Пользователь хочет удалить дисциплину и нажимает на соответствующую кнопку	нет	Открытие окна с удалением дисциплины. Блокировка основного окна.	да

Табл. 14. Тестирование окна с добавлением дисциплины

№ п/п	Описание ситуации	Входные данные	Выходные данные	Тест пройден?
1.	Пользователь заполнил все поля и нажимает на кнопку «Добавить»	Поля формы добавления	Список дисциплин в файле с новой добавленной	Да
2.	Пользователь не заполнил или частично заполнил все поля в форме и нажимает кнопку «Добавить»	Поля формы добавления	Окно с ошибкой о неверно заполненных полях	Да

Табл. 15. Тестирование окна с удалением дисциплины

№ п/п	Описание ситуации	Входные данные	Выходные данные	Тест пройден?
1.	Пользователь нашел и выбрал дисциплину, которую хочет удалить, и нажал кнопку удалить.	Список найденных дисциплин	Список дисциплин в файле без выбранной.	Да
2.	Пользователь нашел и НЕ выбрал дисциплину, которую хочет удалить, и нажал кнопку удалить.	Список найденных дисциплины	Сообщение об ошибке и продолжение работы окна	Да

Табл. 16. Тестирование окна с редактированием дисциплины

№ п/п	Описание ситуации	Входные данные	Выходные данные	Тест пройден?
1.	Пользователь заполнил все поля и нажимает на кнопку «Сохранить»	Поля формы редактирования	Список дисциплин в файле с отредактированной	Да
2.	Пользователь не заполнил или частично заполнил все поля в форме и нажимает кнопку «Сохранить»	Поля формы редактирования	Окно с ошибкой о неверно заполненных полях	Да

Табл. 17. Тестирование окна с поиском дисциплины

№ п/п	Описание ситуации	Входные данные	Выходные данные	Тест пройден?
1.	Пользователь вводит название дисциплины, которую хочет найти и нажимает кнопку «Найти»	String^ строка названия дисциплины	Вывод в listbox дисциплин, подходящих под критерий поиска	Да
2.	Пользователь НЕ вводит название дисциплины, которую хочет найти и нажимает кнопку «Найти»	Нет	Окно с ошибкой о неверно заполненных полях	Да

6. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

6.1 Основное окно приложения

При запуске приложения, пользователь увидит основное окно, на котором и будет выводиться вся информация. Пользователь может закрыть приложение, нажав на «крестик» в правом верхнем углу экрана, перетаскивать окно по экрану, зажав кнопку на верхней части приложения и перетаскивая мышь, но не может изменять размер приложения.

Пользователю доступно 10 кнопок взаимодействия с приложением, на которые он может нажать. Каждая кнопка отвечает за свою задачу и предоставляет пользователю возможности для работы с приложением.

Кнопка «Найти дисциплину» открывает пользователю окно, в котором он может выбрать из выпадающего списка параметры, по которым можно искать дисциплины. При выборе соответствующего параметра (для выбора требуется нажать на этот параметр) и заполнения критерия поиска будет предложен список подходящих под заданные свойства дисциплины.
В поле «оценка» должно быть написано целое число от 0 до 20, а в поле рейтинг – от 0 до 100 (не обязательно целое). Эти поля заполняются аналогично и в других окнах.

Кнопка «Добавить дисциплину» открывает окно, где пользователь может создать новый экземпляр дисциплины с заданными в специальных полях данными. Предусмотрена возможность отмены действия, тогда не произойдёт никаких изменений.

Кнопка «Удалить дисциплину» открывает пользователю окно, в котором он может выбрать из выпадающего списка параметры, по которым можно найти дисциплину, которую пользователь хочет удалить.

Кнопка «Редактировать дисциплину» открывает пользователю окно, в котором он может выбрать из выпадающего списка параметры, по которым можно найти дисциплину, которую пользователь хочет редактировать.

6.2 Добавление или редактирование дисциплины

В каждом из окон редактирования и добавления дисциплины есть поля. В окне добавления все что нужно, это заполнить все поля (над каждым полем есть пояснение, какую информацию нужно в него записать). В поля «название дисциплины» и «темы изучения» необходимо написать текстовую информацию. В поле «семестр изучения» нужно ввести целое число, большее 0, а в поле «рейтинг» число от 0 до 100 (например: «10», «93.85»). После этого нужно нажать кнопку «Добавить», и дисциплина добавится в файл.

В окне редактирования все то же самое, но прежде чем редактировать дисциплину, ее нужно сначала найти и выбрать. Поиск осуществляется путем выбора критерия поиска (для выбора требуется нажать на этот параметр) и его параметра. После заполнения поля параметра немного ниже этого поля будет предложен список дисциплин, которые можно отредактировать. По нажатии кнопки «Сохранить» дисциплина будет изменена.

6.3 Поиск дисциплины по разным критериям

В окне поиска будет тот же самый алгоритм поиска, что и в окне редактирования. Только тут требуется еще нажать на кнопку «Искать».

6.4 Окно удаления дисциплины

Также как и в окне редактирования, прежде чем удалить дисциплину, ее нужно сначала найти. После поиска выбирается дисциплина и нажимается кнопка «Удалить», и дисциплина больше не будет существовать в файле.

ЗАКЛЮЧЕНИЕ

Данная работа посвящена разработке оконного приложения Электронная зачетная книжка с использованием языка C++. Для хранения данных использованы объекты класса List<Discipline^>^, являющегося наследником класса List, приведенного в задании.

В ходе выполнения работы были пройдены все основные этапы разработки программного обеспечения: анализ, написание спецификации, проектирование, разработка алгоритмов, кодирование, тестирование и сопровождение. Проведено автономное тестирование основных методов разработанного класса и комплексное тестирование программы в целом. В результате тестирования ошибок не обнаружено. В данных пунктах описана проделанная в каждом случае работа и ее результаты.

В дальнейшем в приложении может быть добавлена функция автоматического заполнения оценок, загружаемых отчетом.

ЛИТЕРАТУРА

1. Панюкова Т.А., Панюков А.В. Языки и методы программирования: Путеводитель по языку C++. Учебное пособие. – М.: Книжный дом «Либроком», 2013. – 216 с.

2. Макаровских Т.А., Панюков А.В. Языки и методы программирования: Создание простых GUI-приложений с помощью Visual C++: Учебное пособие. Изд. 2-е. – М.: «Ленанд», 2018. – 144 с.

3. Панюкова Т.А. Документирование программного обеспечения: В помощь техническому писателю: Учебное пособие. – М.: Книжный дом «Либроком», 2012. – 264 с.

4. Рязанова, Н. Ю. Программирование на языке C++ в среде Visual Studio CLR Windows Forms : учебное пособие / Н. Ю. Рязанова, К. Л. Тассов, М. В. Филиппов. — Москва : МГТУ им. Н.Э. Баумана, 2017. — 64 с. — ISBN 978-5-7038-4563-9. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/103509 (дата обращения: 28.05.2021). — Режим доступа: для авториз. пользователей.

5. Пахомов Б. И. C/C++ и MS Visual C++ 2012 для начинающих. — СПб.: БХВ-Петербург, 2013. – 512 с. https://karsu.uz/wp-content/uploads/2019/07/7.pdf