Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Математическое обеспечение и применение ЭВМ»

Пояснительная записка к курсовому проекту по дисциплине «Объектно-ориентированное программирование».

Разработка программы с использованием объектно-ориентированного подхода.
ИС “Водительские курсы”

Автор работы:
студенты группы 20ВП1
Ульченко Ф.С.

Принял:
к.т.н. Афонин А.Ю.

Работа сдана _____
Оценка _____

Пенза 2022 г.

Оглавление

Введение

Информационная система (ИС) — совокупность технических и программных средств, организационных методик и персонала, предназначенная для сбора, хранения и передачи данных.

Развитие информационных систем в современном понимании началось в 50-е годы прошлого века. В связи с бурным развитием науки и техники, расширением административного аппарата и его проникновением во все направления деятельности человека, увеличивался объем производимой информации. Появился запрос на автоматизацию и структурирование процесса обработки данных.

Учитывая, что человек производит огромное количество данных во всех сферах своей деятельности, можно с уверенностью сказать, что информационные системы применяются везде. Начиная с бухгалтерской отчетности и поисковика «Google», заканчивая разработками в области искусственного интеллекта и исследованием космоса.

На данном этапе проникновение информационных систем в повседневную жизнь людей практически полное. И их значение для людей сложно переоценить. Каждый сайт, каждое мобильное приложение, интернет в целом — все это является примером ИС.

Профессии, связанные с информационными системами, в последние два десятка лет входят в список самых высокооплачиваемых и престижных. Во всех технических и многих общих высших учебных заведениях есть соответствующая специальность. [1]

1. Постановка задачи и анализ требований

1.1. Анализ требований

1.1.1. Требования к интерфейсу пользователя

Приложение должно предоставлять пользователю удобный интерфейс, позволяющий:

- Добавлять/удалять учащихся;
- Фильтровать по категории;
- Сортировать в алфавитном порядке;
- Выводить результаты предварительных экзаменов, результат сдачи, учет ошибок, посещаемость;
- Выводить список учащихся;

1.1.2. Требования к программным средствам

Анализ задания на разработку позволяет выделить следующие варианты использования (рисунок 1).

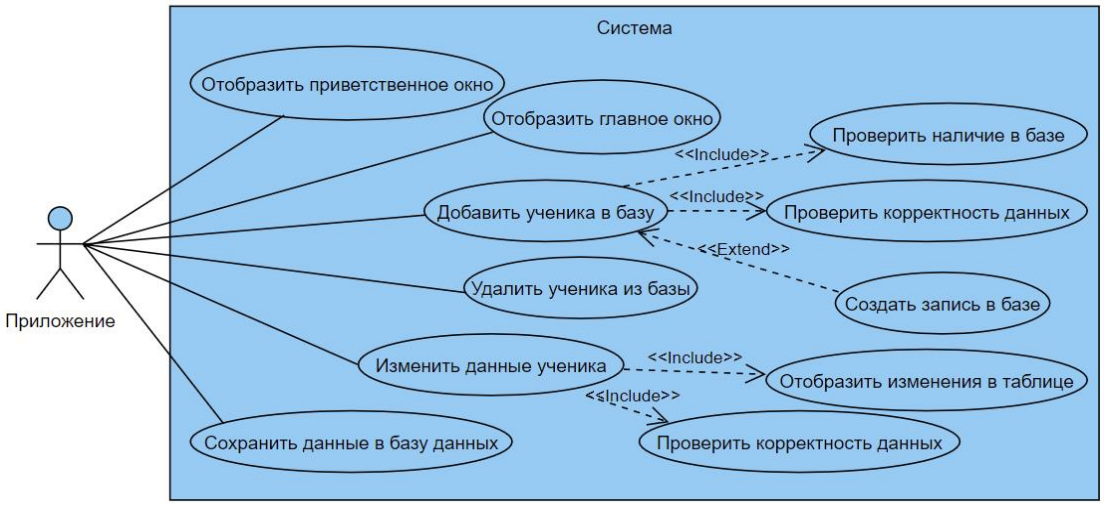

Рисунок 1 – Диаграмма вариантов использования

Описание спецификации прецедента представлена в таблицах 1, 2.

Таблица 1 – Сценарий варианта использования «Отобразить приветственное окно»

Наименование: Отобразить приветственное окно
ID: 1
Краткое описание: на экран выводится форма с основной информацией о приложении
Действующие лица: приложение
Предусловие: приложение запущено
Основной поток: 1. К кнопке добавляется слушатель событий 2. На экран выводится приветственная форма 3. По нажатию на кнопку форма закрывается
Постусловие: приветственное окно закрыто

Таблица 2 – Сценарий варианта использования «Изменить данные ученика»

Наименование: Изменить данные ученика
ID: 5
Краткое описание: изменяются данные определенного ученика
Действующие лица: приложение
Предусловие: на форме прожата кнопка “изменить”
Основной поток: 1. Поиск необходимого ученика 2. Проверка корректности данных 3. Если все данные корректны, и пользователь есть в бд 3.1.Данные пользователя принимают значения из формы 3.2.Изменения данных отображаются в таблице 4. Иначе 4.1.Сообщение об ошибке
Постусловие: данные в таблице обновлены

1.2. Технология разработки программного обеспечения

Технология разработки программного обеспечения представляет собой инженерный подход к разработке программных средств ЭВМ, охватывающий методологию программирования, проблемы обеспечения надежности программ, оценки рабочих характеристик и качества проектов.

Технология разработки программного обеспечения рассматривает вопросы управления проектированием систем программного обеспечения, а также средства и стандарты разработки программ.

1) RUP (Rational Unified Process).

Один из самых известных процессов, использующих итеративную модель разработки – RUP. Он был создан во второй половине 1990-х годов в компании Rational Software. Термином RUP обозначает как методологию, так и продукт компании IBM (ранее Rational) для управления процессом разработки.

Методология RUP описывает абстрактный общий процесс, на основе которого организация или проектная команда должна создать специализированный процесс, ориентированный на ее потребности.

Основные характеристики:

- разработка требований, для описания требований в RUP используются прецеденты использования (use cases). Полный набор прецедентов использования системы вместе с логическими отношениями между ними называется моделью прецедентов использования. Каждый прецедент использования – это описание сценариев взаимодействия пользователя с системой, полностью выполняющего конкретную пользовательскую задачу. Согласно RUP все функциональные требования должны быть представлены в виде прецедентов использования.

- итеративная разработка, проект RUP состоит из последовательности итераций с рекомендованной продолжительностью от 2 до 6 недель. Перед началом очередной итерации определяется набор прецедентов использования, которые будут реализованы к её завершению.

Жизненный цикл проекта RUP состоит из четырех фаз. Последовательность этих фаз фиксирована, но число итераций, необходимых для завершения каждой фазы, определяется индивидуально для каждого конкретного проекта. Фазы RUP нельзя отождествлять с фазами водопадной модели – их назначение и содержание принципиально различны.

2)Scrum.

Scrum предоставляет эмпирический подход к разработке ПО. Этот процесс быстр, адаптивен, умеет подстраиваться и отличен от каскадной модели. Scrum основан на повторяющихся циклах, это делает его более гибким и предсказуемым.

Для начала определим роли, которые участвуют в процессе:

Scrum мастер (Scrum Master), Владелец продукта (Product Owner), Команда (Team).

Scrum Мастер - самая важная роль в методологии. Scrum Мастер отвечает за успех Scrum в проекте. Как правило, эту роль в проекте играет менеджер проекта или лидер команды (Team Leader). Важно подчеркнуть, что Scrum Мастер не раздает задачи членам команды. В Scrum команда является самоорганизующейся и самоуправляемой.

Основные обязанности Scrum Мастера таковы:

− создает атмосферу доверия,

− участвует в митингах в качестве фасилитатора - человека, обеспечивающий успешную групповую коммуникацию

− устраняет препятствия − делает проблемы и открытые вопросы видимыми

− отвечает за соблюдение практик и процесса в команде

Scrum Мастер отслеживает прогресс команды при помощи Sprint Backlog, отмечая статус всех задач в спринте.

3)Crystal Clear.

Легковесная гибкая методология, созданная Алистером Коуберном, которая предназначена для небольших команд в 6-8 человек для разработки некритичных бизнес-приложений. Как и все гибкие методологии, Crystal Clear больше опирается на людей, чем на процессы 38 и артефакты.

Crystal Clear использует семь методов/практик, три из которых являются обязательными:

- частая поставка продукта;
- улучшения через рефлексию;
- личные коммуникации;
- чувство безопасности;
- фокусировка;
- простой доступ к экспертам;
- качественное техническое окружение.

4)RAD – для разработки интерфейса

RAD (англ. rapidapplicationdevelopment — быстрая разработка приложений) — концепция создания средств разработки программных продуктов, уделяющая особое внимание быстроте и удобству программирования, созданию технологического процесса, позволяющего программисту максимально быстро создавать компьютерные программы.

RAD предполагает, что разработка ПО осуществляется небольшой командой разработчиков за срок порядка трёх-четырёх месяцев путём использования инкрементного прототипирования с применением инструментальных средств визуального моделирования и разработки. Технология RAD предусматривает активное привлечение заказчика уже на ранних стадиях — обследование организации, выработка требований к системе.

Последнее из указанных свойств подразумевает полное выполнение требований заказчика как функциональных, так и нефункциональных, с учётом их возможных изменений в период разработки системы, а также получение качественной документации, обеспечивающей удобство эксплуатации и сопровождения системы. Это означает, что дополнительные затраты на сопровождение сразу после поставки будут значительно меньше. Таким образом, полное время от начала разработки до получения приемлемого продукта при использовании этого метода значительно сокращается. [2]

В результате изучения технологий разработки программного продукта, была выбрана технология RAD по следующим причинам:

- интерфейс, устраивающий пользователя;
- легкая адаптируемость проекта к изменяющимся требованиям; быстрота продвижения программного продукта

2. Проектирование программы

2.1. Модель интерфейса

В результате проектирования программы были реализованы формы, представленные на рисунках 2-3.

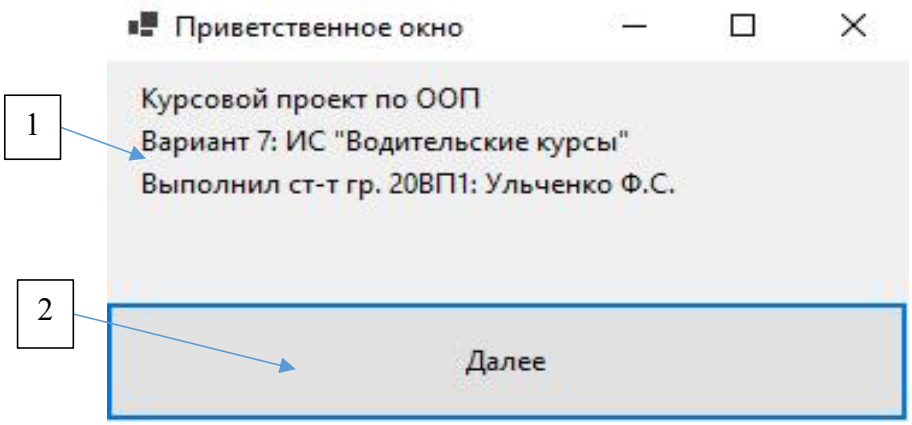

Рисунок 2 – Приветственное окно

На рисунке 2 приведены следующие обозначения:

1) Информация о программе

2) Кнопка “Далее” (переход на главную страницу)

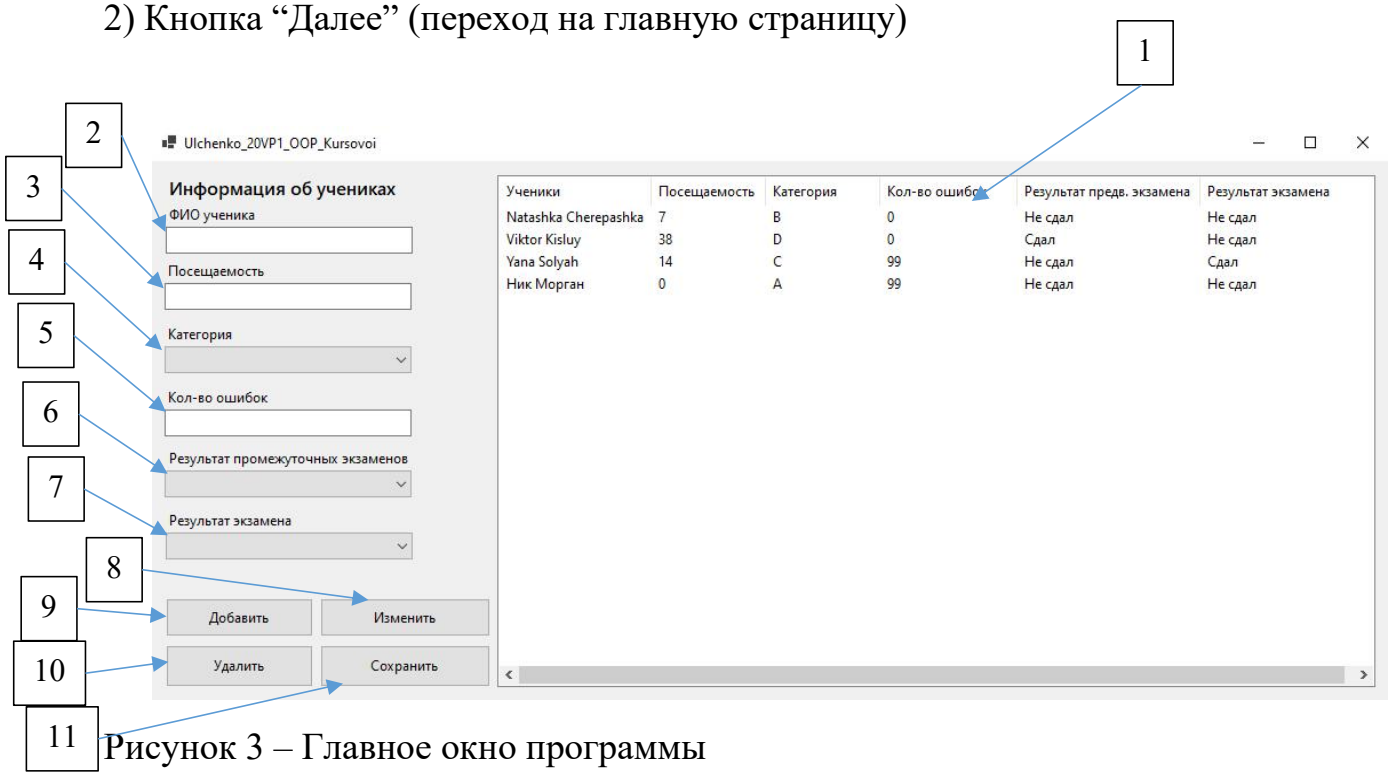

Рисунок 3 – Главное окно программы

На рисунке 3 приведены следующие обозначения:

1. – таблица для вывода базы данных,

2. – поле для ввода ФИО ученика,

3. – поле для ввода посещаемости,

4. – выпадающий список для выбора категории прав,

5. – поле для ввода количества ошибок,

6. – выпадающий список для выбора результатов промежуточного экзамена,

7. – выпадающий список для выбора результатов экзамена,

8. – кнопка для изменения данных об ученике,

9. – кнопка для добавления ученика в базу,

10. – кнопка для удаления ученика из базы,

11. – кнопка для сохранения базы данных в файл.

2.2. Структура программного обеспечения

Для создания программы был использован объектно-ориентированный подход. Структура программы в виде диаграммы классов представлена на рисунке 4.

На диаграмме классов представлены все созданные классы на основе которых была реализована ИС “Водительские курсы”.

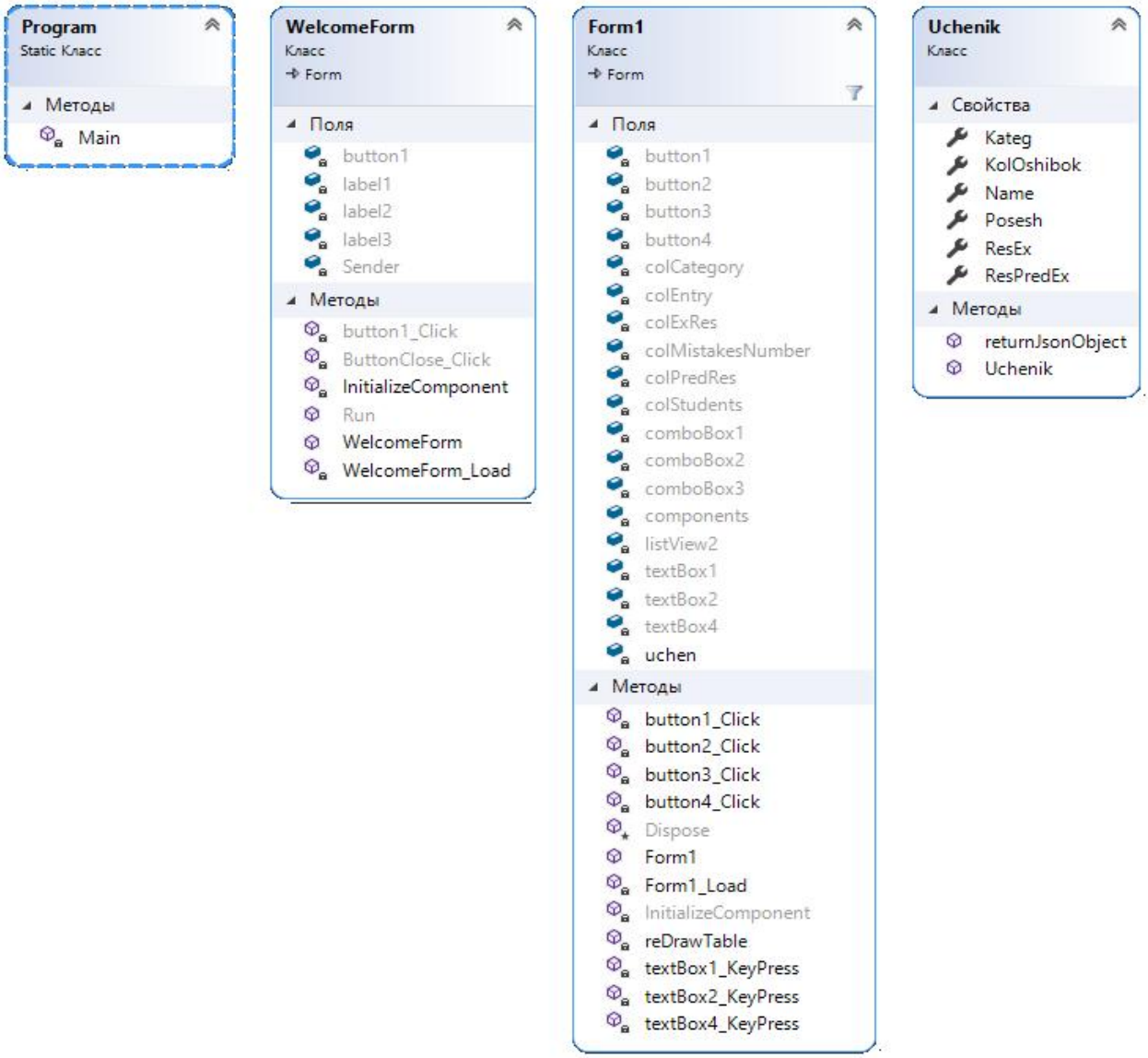

Рисунок 4 – Диаграмма классов

Описание компонент диаграммы классов:

1) Класс Program – хранит метод Main(входная точка программы).

2) Класс WelcomeForm– приветственное окно с основной информацией о программе.

3) Класс Form1– главное окно программы.

4) Класс Uchenik обладает свойствами для хранения информации об ученике.

Описание полей и методов классов представлено в виде комментариев к коду в приложении А.

3. Реализация программы

Для решения поставленной задачи было написано приложение в среде Visual Studio 2019 с использованием языка C#, а весь интерфейс реализован в WinForms. Исходный код приведен в приложении А. Скриншоты программы представлены в руководстве пользователя.

Была составлена диаграмма компонентов, которая отображает разбиение программной системы на структурные компоненты и связи между компонентами (Рисунок 5.).

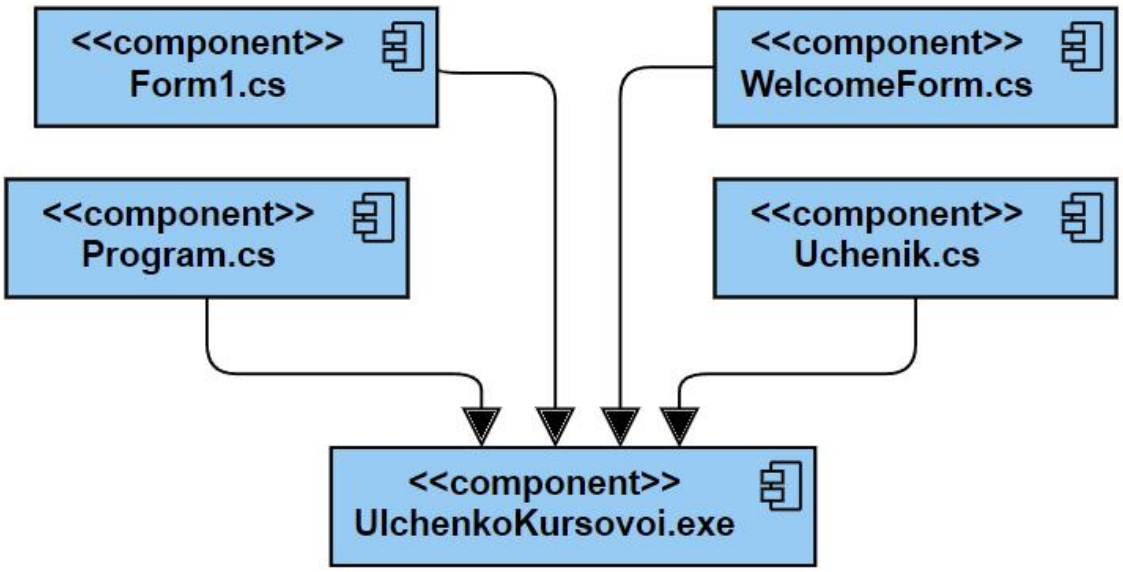

Рисунок 5 – Диаграмма компонентов

В таблице 3 представлено описание компонент.

Таблица 3 – Компоненты приложения

Компонент	Назначение
UlchenkoKursovoi.exe	Исполняемый файл проекта
WelcomeForm.cs	Приветственное окно
Form1.cs	Главное окно программы

Продолжение таблицы 3

Компонент	Назначение
Uchenik.cs	Класс для хранения информации об ученике
Program.cs	Исходный файл

4. Тестирование программы

4.1. Виды тестирования

Существуем много видов тестирования программного продукта. Были рассмотрены следующие виды тестирования:

1) Функциональное тестирование.

Функциональное тестирование — это тестирование ПО в целях проверки реализуемости функциональных требований, то есть способности ПО в определённых условиях решать задачи, нужные пользователям. Функциональное тестирование на сегодняшний день является одним из наиболее часто применяемых видов тестирования, так как позволяет наиболее полно проверить программу на соответствие требованиям.

2) Нефункциональное тестирование.

Нефункциональное тестирование позволяет проверить соответствие свойств программного обеспечения с поставленными нефункциональными требованиями.

3) Модульное тестирование

Каждая сложная программная система состоит из отдельных частей – модулей, выполняющих ту или иную функцию в составе системы. Для того, чтобы удостовериться в корректной работе всей системы, необходимо вначале протестировать каждый модуль системы по отдельности. В случае возникновения проблем при тестировании системы в целом это позволяет проще выявить модули, вызвавшие проблему, и устранить соответствующие дефекты в них.

4) Юзабилити-тестирование

Юзабилити-тестирование — исследование, выполняемое с целью определения, удобен ли некоторый искусственный объект (такой как веб страница, пользовательский интерфейс или устройство) для его предполагаемого применения. Таким образом, проверка эргономичности измеряет эргономичность объекта или системы.

Было принято решение осуществить проверку программы с помощью функционального тестирования. Такой выбор обусловлен тем, что к программе были предъявлены только функциональные требования, выполнение которых нужно проверить

4.2. Функциональное тестирование

В курсовом проекте было проведено функциональное тестирование, результаты которого представлены в таблице 4.

Таблица 4 – Функциональные тесты

Состав теста	**Ожидаемый результат**	**Наблюдаемый результат**
Добавление ученика с пустыми полями	Сообщение об ошибке	Тест пройден(Рисунок 6)
Удаление ученика которого нет в базе	Сообщение об ошибке	Тест пройден(Рисунок 7)
Добавление пользователя, который уже есть в базе данных	Сообщение об ошибке	Тест пройден(Рисунок 8)
Добавление пользователя	Добавленный пользователь отобразится в таблице	Тест пройден(Рисунок 9)

Продолжение таблицы 4

Состав теста	**Ожидаемый результат**	**Наблюдаемый результат**
Удаление пользователя	Удаленный пользователь перестанет отображаться в таблице	Тест пройден(Рисунок 10)

Ниже представлены результаты тестирования на рисунках 6-10.

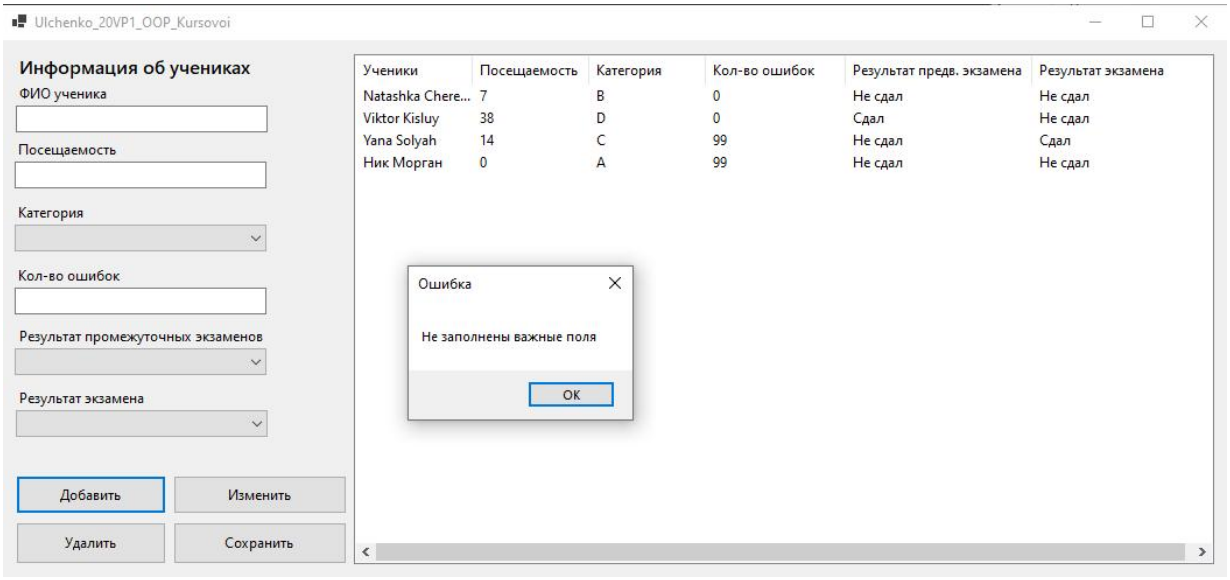

Рисунок 6 – Добавление ученика с пустыми полями

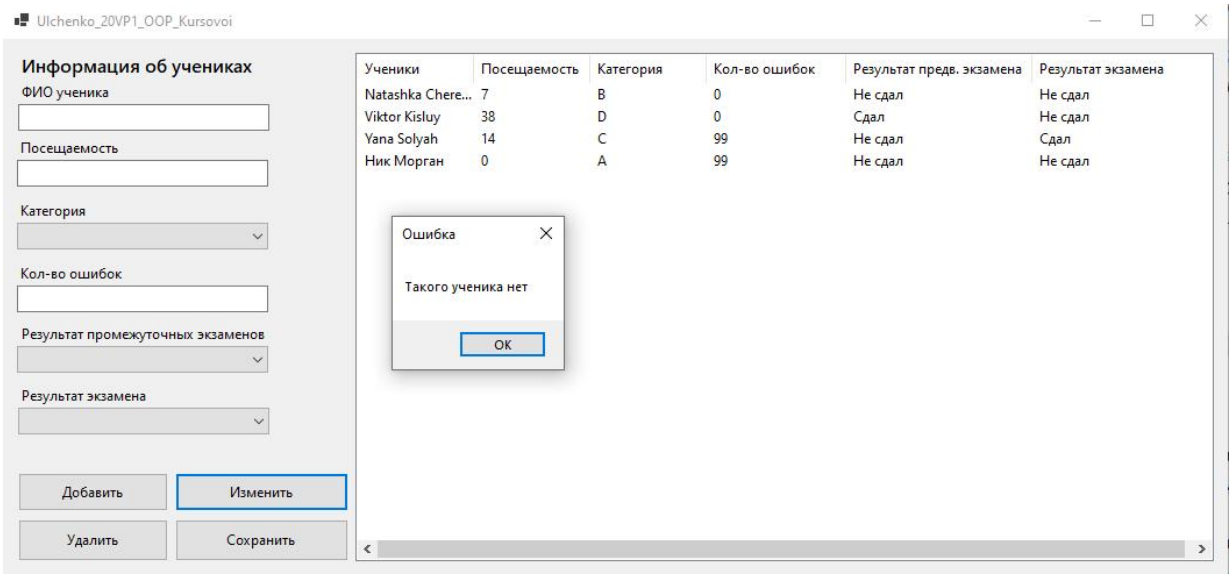

Рисунок 7 – Удаление ученика которого нет в базе

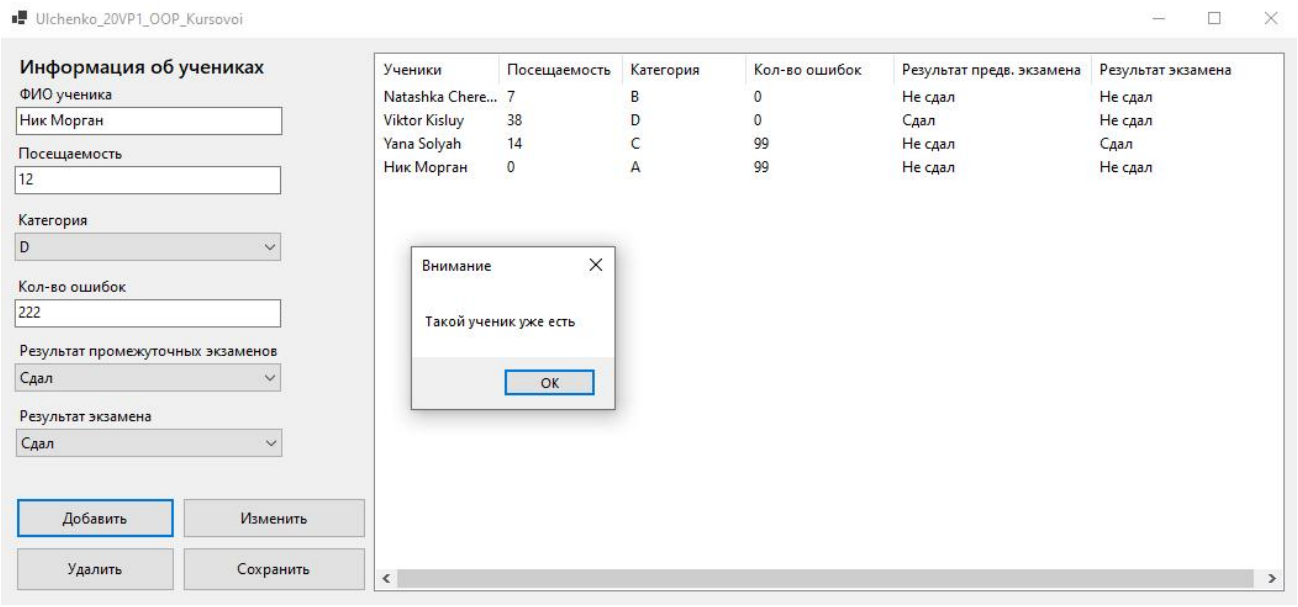

Рисунок 8 – Добавление пользователя, который уже есть в базе данных

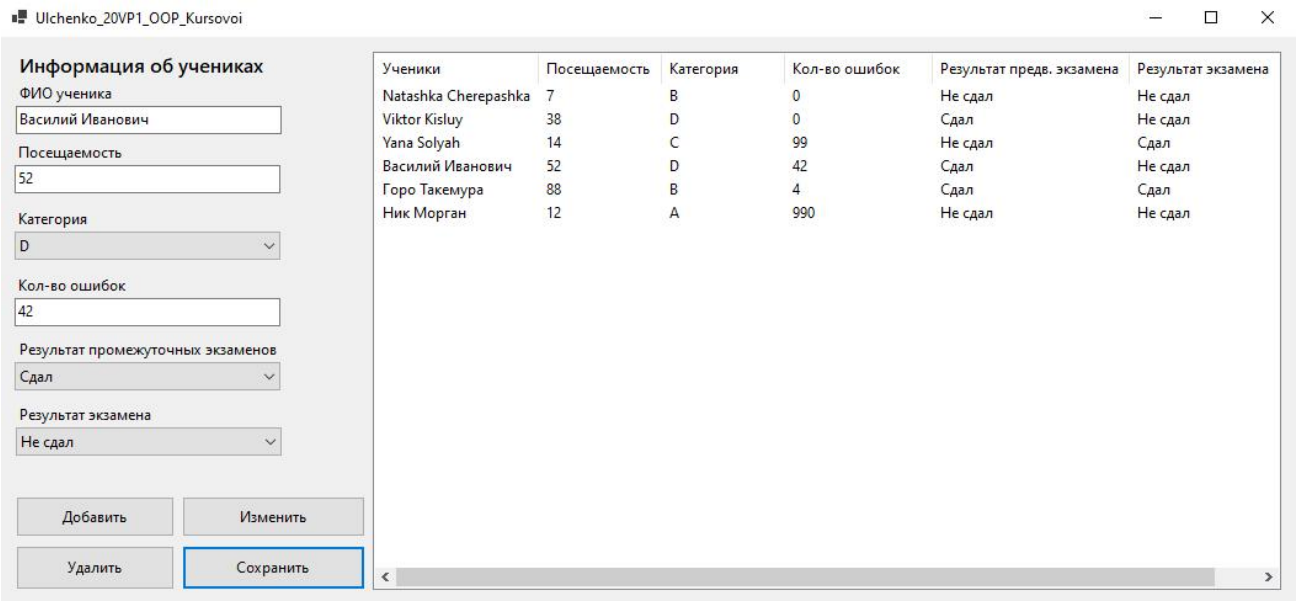

Рисунок 9 – Добавление пользователя

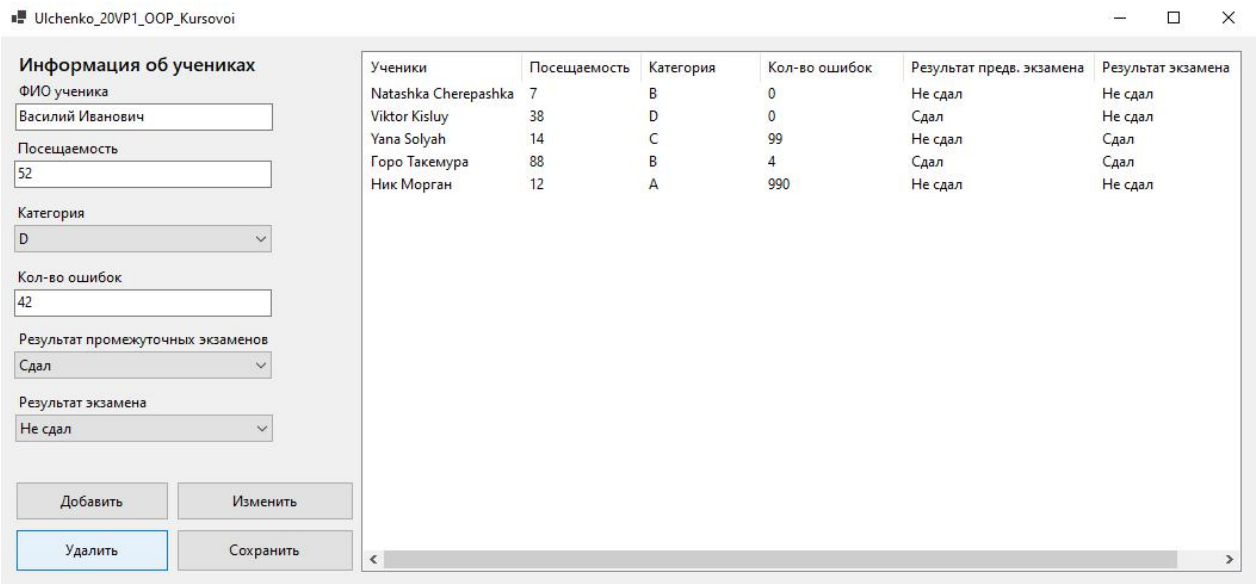

Рисунок 10 – Удаление пользователя

В ходе выполнения тестирования несовпадения ожидаемого результата не выявлены. Следовательно, можно сделать вывод, что программа работает корректно.

5. Руководство пользователя

После запуска программы появится приветственное окно с основной информацией о программе. Для продолжения необходимо кликнуть по кнопке “Далее”(Рисунок 11.).

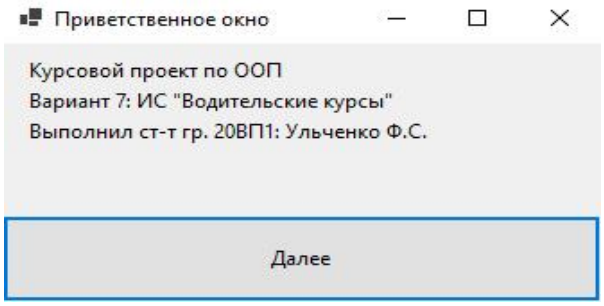

Рисунок 11 – Приветственное окно

После закрытия приветственного окна откроется основная форма программы(Рисунок 12.). При вместе с открытием главного окна программа считывает Для отображения информации об учениках, хранящейся в базе данных, используется таблица. Кнопка “Удалить” удаляет из базы ученика, ФИО которого находится в соответствующем поле. Кнопки “Добавить” и ”Изменить” добавляют нового ученика в базу, изменяют существующего соответственно, беря необходимую информацию из полей в левой части экрана. Кнопка “Сохранить” сохраняет текущее состояние базы данных в файл.

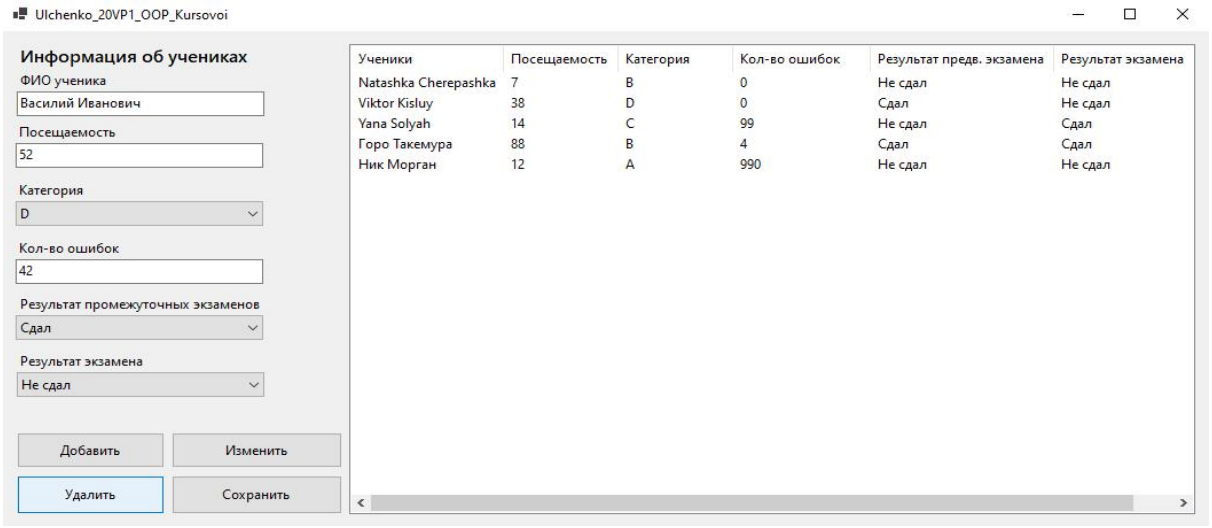

Рисунок 12 – Главное окно программы

Заключение

В рамках данной курсовой работы были выполнены все поставленные задачи:

• Составлены требования к программе

• Спроектирована программа

• Реализована программа

• Протестирована программа

• Проведено сравнение эффективности реализованных методов и классов

• Проанализированы результаты

Результатом курсовой работы является программа, которая позволяет:

- Добавлять/удалять учащихся;
- Фильтровать по категории;
- Сортировать в алфавитном порядке;
- Выводить список учащихся, их результаты предварительных экзаменов, результат сдачи, учет ошибок, посещаемость;

Список литературы

1) Информационные системы – Режим доступа: https://wiki.fenix.help/informatika/informatsionnaya-sistema (Дата обращения: 29.03.2022)

2) Гибкие методологии разработки | Habr.com – Режим доступа: https://habr.com/ru/company/it-guild/blog/341924/ (Дата обращения: 30.03.2022)

3) System.Text.Json Пространство имен | Microsoft Docs – Режим доступа: https://docs.microsoft.com/ru-RU/dotnet/api/system.text.json?view=net-6.0 (Дата обращения: 12.04.2022)

4) Работа с JSON | Metanit.com – Режим доступа: https://metanit.com/sharp/tutorial/6.5.php (Дата обращения: 21.04.2022)

5) Васильев А.С#. Объектно-ориентированное программирование: Учебный курс. – СПб.: Питер, 2012. – 320 с.: ил

Приложение А

Файл Program.cs

```
using System;
using System.Windows.Forms;

namespace UlchenkoKursovoi
{
    static class Program
    {
        [STAThread]
        static void Main()
        {
            Application.SetHighDpiMode(HighDpiMode.SystemAware);
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new WelcomeForm());
            Application.Run(new Form1());
        }
    }
}
```

Файл Uchenik.cs

```
namespace UlchenkoKursovoi
{
    class Uchenik
    {
        /// <summary>
        /// Имя ученика
        /// </summary>
        public string Name { get; set; }
        /// <summary>
        /// Категория прав
        /// </summary>
        public string Kateg { get; set; }
        /// <summary>
        /// Колличество посещений
        /// </summary>
        public int ?Posesh { get; set; }
        /// <summary>
        /// Колличество ошибок
        /// </summary>
        public int ?KolOshibok { get; set; }
        /// <summary>
        /// Результаты предварительных экзаменов
        /// </summary>
        public string ResPredEx { get; set; }
        /// <summary>
        /// Результаты финального экзамена
        /// </summary>
        public string ResEx { get; set; }
        /// <summary>
        /// Класс "ученик автошколлы"
        /// </summary>
        /// <param name="name">Имя ученика</param>
        /// <param name="kateg">Категория прав</param>
        /// <param name="posesh">Колличество посещений</param>
        /// <param name="kolOshibok">Колличество ошибок</param>
        /// <param name="resPredEx">Результаты предварительных экзаменов</param>
        /// <param name="resEx">Результаты финального экзамена</param>
        public Uchenik(string name, string kateg, int posesh, int kolOshibok, string
resPredEx, string resEx)
        {
            Name = name;
```

```
            Kateg = kateg;
            Posesh = posesh;
            KolOshibok = kolOshibok;
            ResPredEx = resPredEx;
            ResEx = resEx;
        }
        /// <summary>
        /// Формирование строки для json из всех свойств класса
        /// </summary>
        /// <returns>Строка готовая для записи в json файл</returns>
        public string returnJsonObject()
        {
            return "{ 'Name': '" + Name + "', 'Kateg': '" + Kateg + "','Posesh': '" + Posesh
+
                "','KolOshibok':'" + KolOshibok + "','ResPredEx':'" + ResPredEx+
"','ResEx':'" + ResEx + "'}";
        }
    }
}
```

Файл WelcomeForm.cs

```
using System;
using System.Drawing;
using System.Windows.Forms;

namespace UlchenkoKursovoi
{
    public partial class WelcomeForm : Form
    {
        public WelcomeForm()
        {
            InitializeComponent();
        }

        private void InitializeComponent()
        {
            this.button1 = new System.Windows.Forms.Button();
            this.label1 = new System.Windows.Forms.Label();
            this.label2 = new System.Windows.Forms.Label();
            this.label3 = new System.Windows.Forms.Label();
            this.SuspendLayout();
            //
            // button1
            //
            this.button1.Dock = System.Windows.Forms.DockStyle.Bottom;
            this.button1.Location = new System.Drawing.Point(0, 72);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(290, 54);
            this.button1.TabIndex = 0;
            this.button1.Text = "Далее";
            this.button1.UseVisualStyleBackColor = true;
            this.button1.Click += new System.EventHandler(this.button1_Click);
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(12, 9);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(148, 15);
            this.label1.TabIndex = 1;
            this.label1.Text = "Курсовой проект по ООП";
            //
            // label2
            //
```

```
            this.label2.AutoSize = true;
            this.label2.Location = new System.Drawing.Point(12, 28);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(210, 15);
            this.label2.TabIndex = 2;
            this.label2.Text = "Вариант 7: ИС \"Водительские курсы\"";
            //
            // label3
            //
            this.label3.AutoSize = true;
            this.label3.Location = new System.Drawing.Point(12, 47);
            this.label3.Name = "label3";
            this.label3.Size = new System.Drawing.Size(229, 15);
            this.label3.TabIndex = 3;
            this.label3.Text = "Выполнил ст-т гр. 20ВП1: Ульченко Ф.С.";
            //
            // WelcomeForm
            //
            this.ClientSize = new System.Drawing.Size(290, 126);
            this.Controls.Add(this.label3);
            this.Controls.Add(this.label2);
            this.Controls.Add(this.label1);
            this.Controls.Add(this.button1);
            this.Name = "WelcomeForm";
            this.Text = "Приветственное окно";
            this.Load += new System.EventHandler(this.WelcomeForm_Load);
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        private void WelcomeForm_Load(object sender, EventArgs e)
        {
            this.MinimumSize = new Size(350, 200);
            this.MaximumSize = new Size(350, 400);
        }
    }
}
```

Файл Form1.cs

```
using Newtonsoft.Json;
using System;
using System.Collections.Generic;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Windows.Forms;

namespace UlchenkoKursovoi
{
    public partial class Form1 : Form
    {
        List<Uchenik> uchen =
JsonConvert.DeserializeObject<List<Uchenik>>(File.ReadAllText("database.json"));

        public Form1()
        {
            InitializeComponent();
            reDrawTable();

            textBox1.KeyPress += new KeyPressEventHandler(textBox1_KeyPress);
            textBox2.KeyPress += new KeyPressEventHandler(textBox2_KeyPress);
            textBox4.KeyPress += new KeyPressEventHandler(textBox4_KeyPress);
```

```
        }
        /// <summary>
        /// Добавление ученика
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void button1_Click(object sender, EventArgs e)
        {
            if ((textBox1.Text == "") || (textBox2.Text == "") || (textBox4.Text == ""))
            {
                MessageBox.Show("Не заполнены важные поля", "Ошибка");
                return;
            }

            var n = textBox1.Text;

            if (uchen.Where(u => u.Name == textBox1.Text).Count() != 0)
            {
                MessageBox.Show("Такой ученик уже есть", "Внимание");
                Uchenik nay = uchen.Where(u => u.Name.ToLower() ==
textBox1.Text.ToLower()).First();
                textBox1.Text = nay.Name;
                textBox2.Text = nay.Posesh.ToString();
                textBox4.Text = nay.KolOshibok.ToString();
                comboBox3.SelectedItem = nay.Kateg;
                comboBox1.SelectedItem = nay.ResPredEx;
                comboBox2.SelectedItem = nay.ResEx;
            }
            else
            {
                var k = comboBox3.Text;
                var p = int.Parse(textBox2.Text);
                var ko = int.Parse(textBox4.Text);
                var rp = comboBox1.Text;
                var re = comboBox2.Text;

                var uch = new Uchenik(
                        name: n,
                        kateg: k,
                        posesh: p,
                        kolOshibok: ko,
                        resPredEx: rp,
                        resEx: re
                    );

                uchen.Add(uch);

                var item1 = new ListViewItem(new[] {
                   n.ToString(),
                   p.ToString(),
                   k.ToString(),
                   ko.ToString(),
                   rp.ToString(),
                   re.ToString()
                });
                listView2.Items.Add(item1);
            }
        }

        /// <summary>
        /// Удаление ученика
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
```

```
        private void button2_Click(object sender, EventArgs e)
        {
            try
            {
                uchen.Remove(uchen.Where(u => u.Name.ToLower() ==
textBox1.Text.ToLower()).First());
                reDrawTable();
            }
            catch
            {
                MessageBox.Show("Такого ученика нет", "Ошибка");
            }
        }

        /// <summary>
        /// Изменение данных об ученике
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void button3_Click(object sender, EventArgs e)
        {
            if (uchen.Where(u => u.Name == textBox1.Text).Count() == 0)
            {
                MessageBox.Show("Такого ученика нет", "Ошибка");
                return;
            }
            try
            {
                uchen.Select(u => u).Where(u => u.Name.ToLower() ==
textBox1.Text.ToLower()).First().Posesh = int.Parse(textBox2.Text);
                uchen.Select(u => u).Where(u => u.Name.ToLower() ==
textBox1.Text.ToLower()).First().Kateg = comboBox3.Text;
                uchen.Select(u => u).Where(u => u.Name.ToLower() ==
textBox1.Text.ToLower()).First().KolOshibok = int.Parse(textBox4.Text);
                uchen.Select(u => u).Where(u => u.Name.ToLower() ==
textBox1.Text.ToLower()).First().ResPredEx = comboBox1.Text;
                uchen.Select(u => u).Where(u => u.Name.ToLower() ==
textBox1.Text.ToLower()).First().ResEx = comboBox2.Text;
                reDrawTable();
            }
            catch
            {
                MessageBox.Show("Одно из полей не заполнено", "Ошибка");
            }
        }

        /// <summary>
        /// Запись в файл
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void button4_Click(object sender, EventArgs e)
        {
            File.WriteAllText("database.json", JsonConvert.SerializeObject(uchen));
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            this.MinimumSize = new Size(1069, 500);
        }

        /// <summary>
        /// Метод отоброжения актуальных данных в таблице
        /// </summary>
```

```
        private void reDrawTable()
        {
            listView2.Items.Clear();

            foreach (var uc in uchen)
            {
                var item1 = new ListViewItem(new[] {
                    uc.Name,
                    uc.Posesh.ToString(),
                    uc.Kateg,
                    uc.KolOshibok.ToString(),
                    uc.ResPredEx,
                    uc.ResEx
                });
                listView2.Items.Add(item1);
            }
        }

        /// <summary>
        /// Запрет на ввод цифр в поле задания имени ученика
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
        {
            char ch = e.KeyChar;

            if (Char.IsDigit(ch) && ch != 8)
            {
                e.Handled = true;
            }
        }

        /// <summary>
        /// Запрет на ввод букв в поле посещаемости
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void textBox2_KeyPress(object sender, KeyPressEventArgs e)
        {
            char ch = e.KeyChar;

            if (!Char.IsDigit(ch) && ch != 8)
            {
                e.Handled = true;
            }
        }

        /// <summary>
        /// Запрет на ввод букв в поле задания колличества ошибок
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void textBox4_KeyPress(object sender, KeyPressEventArgs e)
        {
            char ch = e.KeyChar;

            if (!Char.IsDigit(ch) && ch != 8)
            {
                e.Handled = true;
            }
        }
    }
}
```