Министерство транспорта Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

«Российский университет транспорта»

(ФГАОУ ВО РУТ(МИИТ), РУТ (МИИТ)

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Лабораторная работа № 19

по дисциплине: «Программирование и основы алгоритмизации»

на тему: «Запросы LINQ»

Выполнил: ст. гр. ТУУ-211

Голюдбин А.В.

Вариант №3

Проверил: к.т.н., доц. Сафронов А.И.

1. **Формулировка цели работы**

Освоить навыки подготовки наборов данных по нечётко сформулированному описанию предметной области при учёте особенностей запросов, отклики на которые необходимо предоставлять заявленным (потенциальным) пользователям программного обеспечения.

1. **Описание задачи**

Описание предметной области для подготовки данных. Пользователями являются сотрудники из числа руководящего состава кафедры. Необходима информация о преподавателях и инженерах, работающих на кафедре, а также об их занятости (по научно-исследовательским работам, преподаваемым дисциплинам, организационной работе с обучающимися, методической работе).

Реализовать запросы *LINQ*:

– выдавать сводную информацию обо всех работниках кафедры;

– выдавать информацию о занятости преподавателя научно-исследовательской работой;

– выдавать информацию о занятости преподавателя организационной работой с обучающимися;

– выдавать информацию о занятости преподавателя методической работой;

– выдавать информацию о преподавателе, ведущем указанный вид занятий по указанной дисциплине;

– выдавать информацию о видах занятий, которые проводятся по выбранной из списка (впечатывать наименования запрещено) дисциплине.

1. **Сеть Петри запрограммированного технологического процесса**

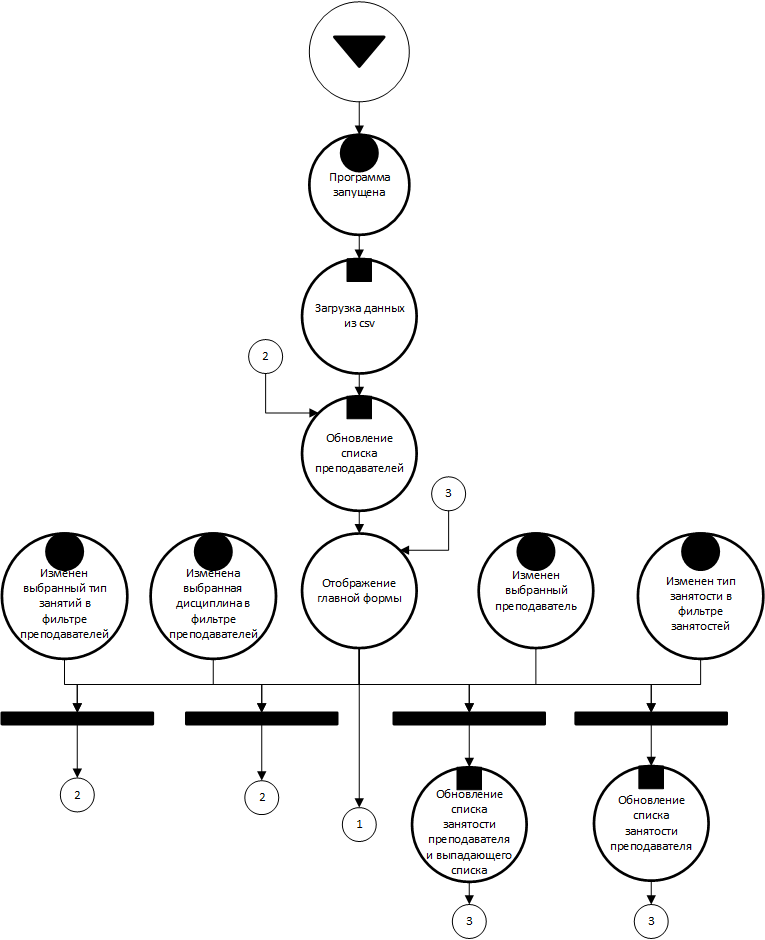


Рисунок 1 – Сеть Петри запрограммированного технологического процесса, часть 1

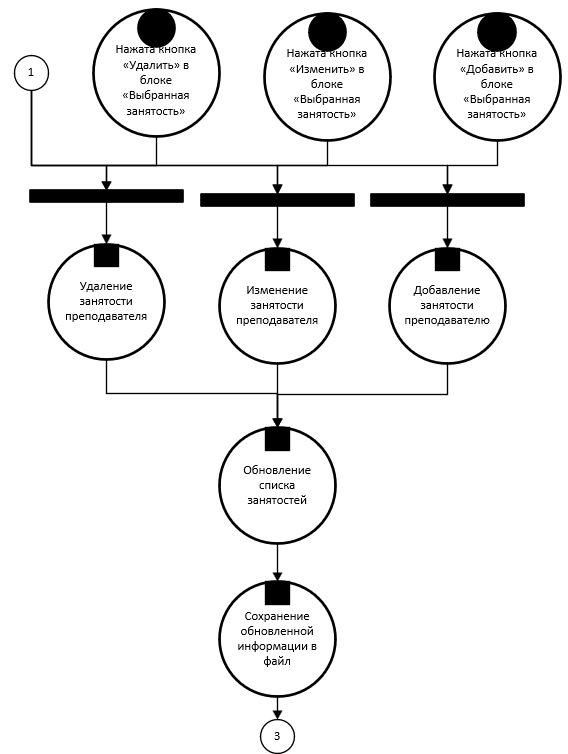


Рисунок 2 – Сеть Петри запрограммированного технологического процесса, часть 2

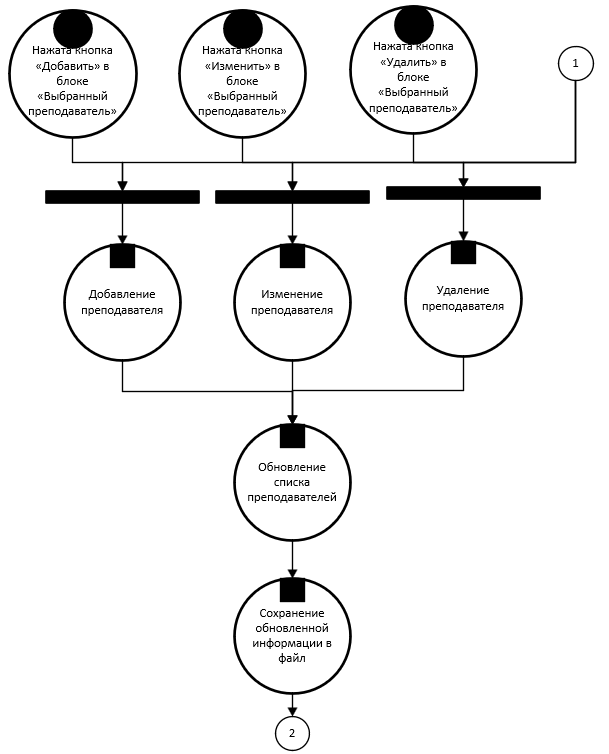


Рисунок 3 – Сеть Петри запрограммированного технологического процесса, часть 3

1. **Схемы алгоритмов методов в составе решения, отмеченных на сети Петри в качестве «эффектов» (метка )**

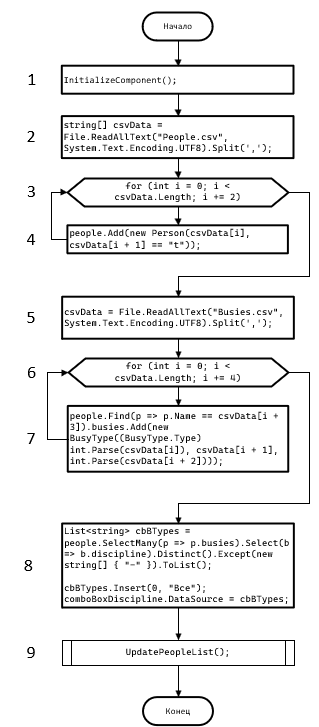


Рисунок 4 – Загрузка данных из *csv*

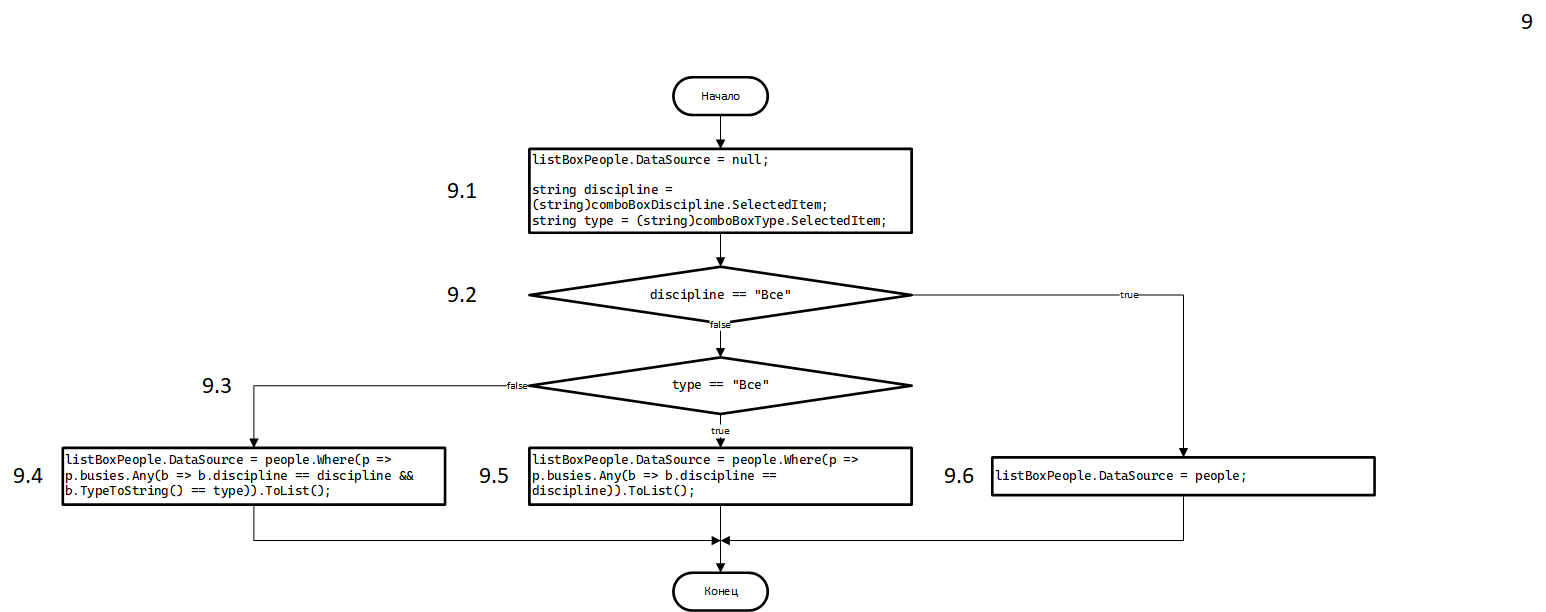


Рисунок 5 – Обновление списка преподавателей (метод UpdatePeople)

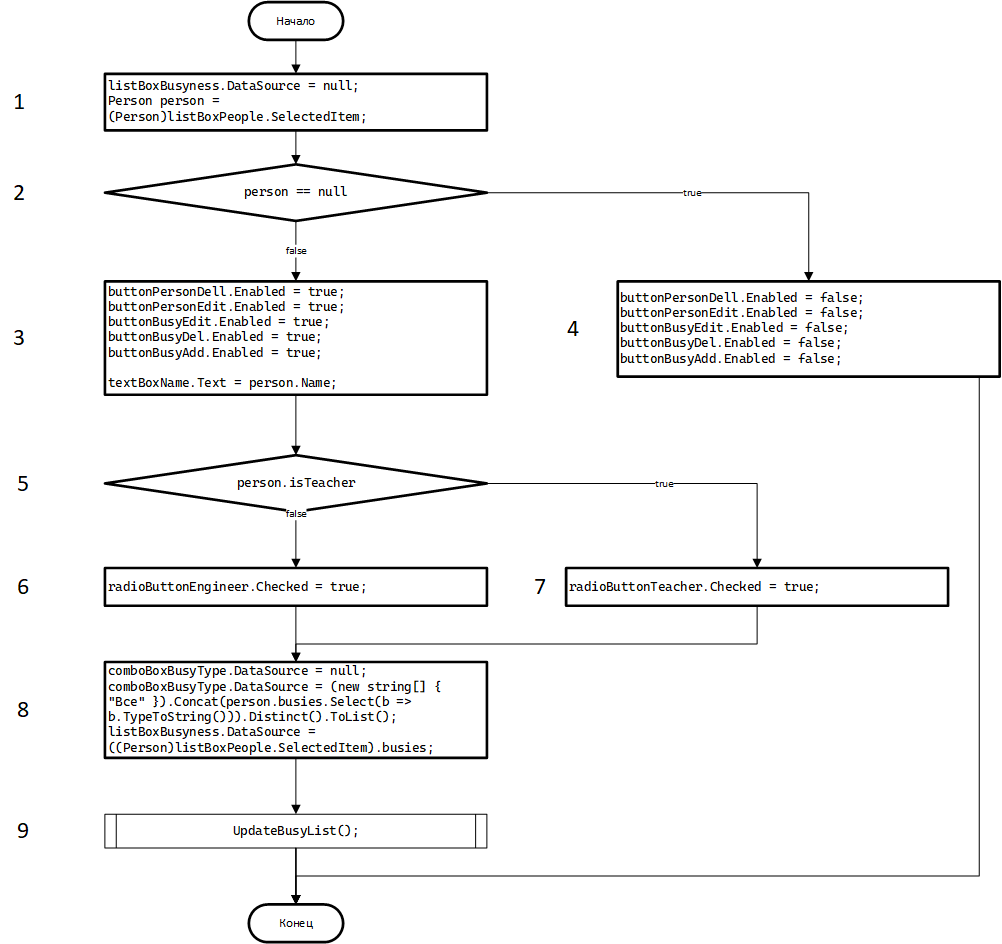


Рисунок 6 – Обновление списка занятости преподавателя и выпадающего списка

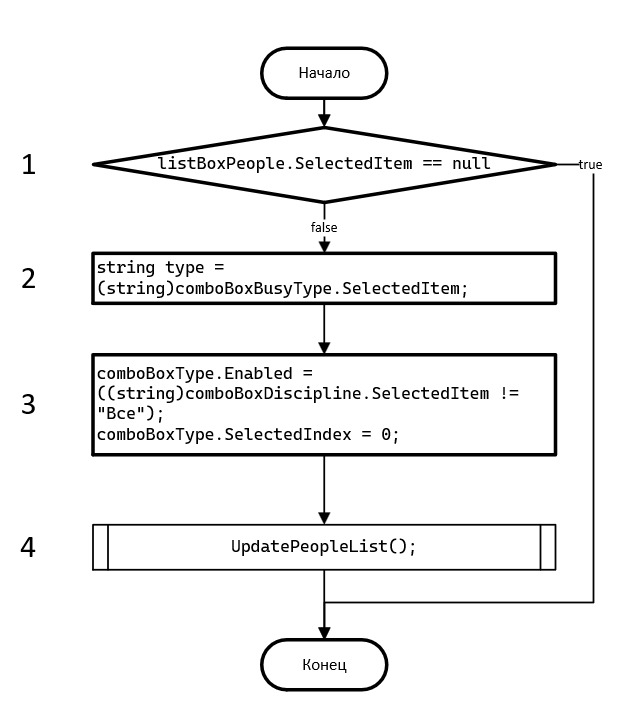


Рисунок 7 – Обновление списка занятости преподавателя

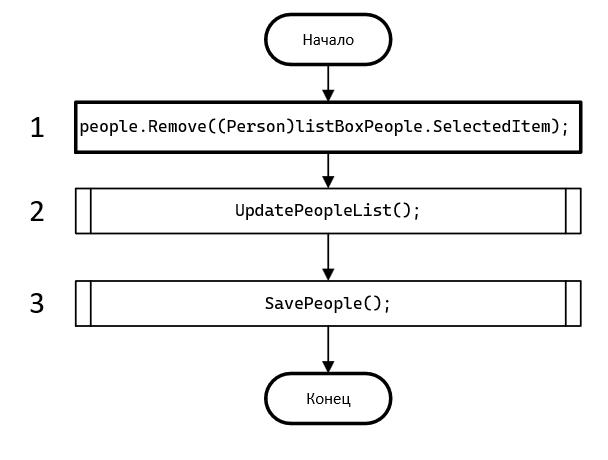


Рисунок 8 – Удаление преподавателя

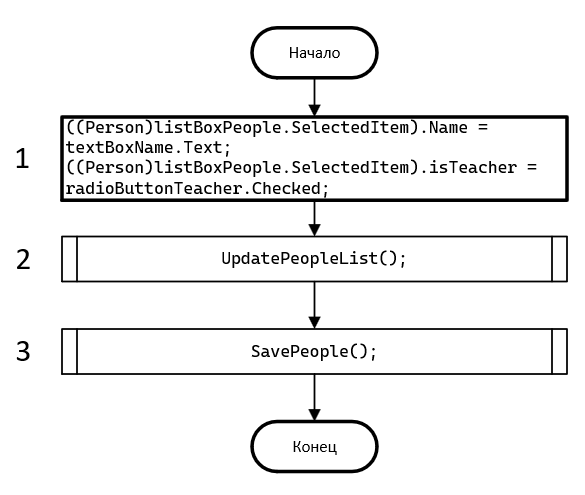


Рисунок 9 – Изменение преподавателя

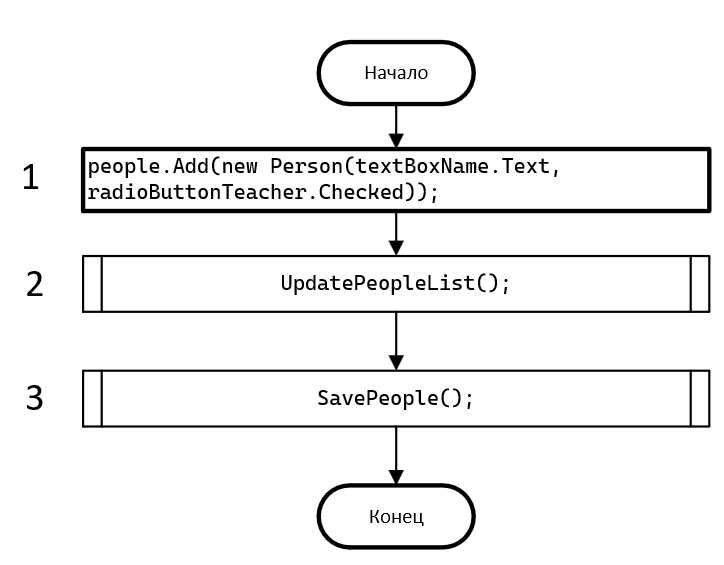


Рисунок 10 – Добавление преподавателя

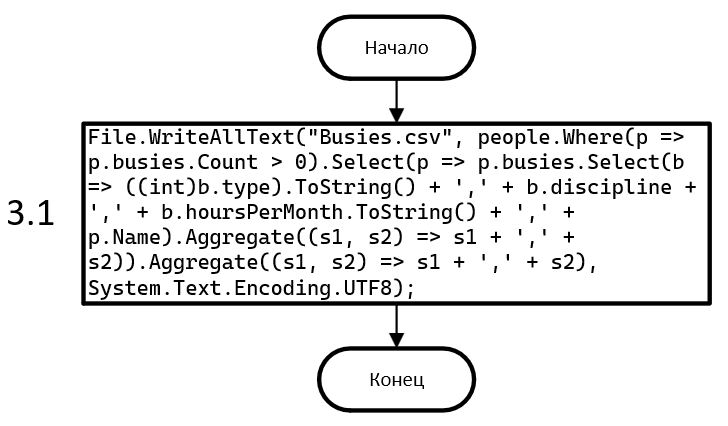


Рисунок 11 – Метод SavePeople

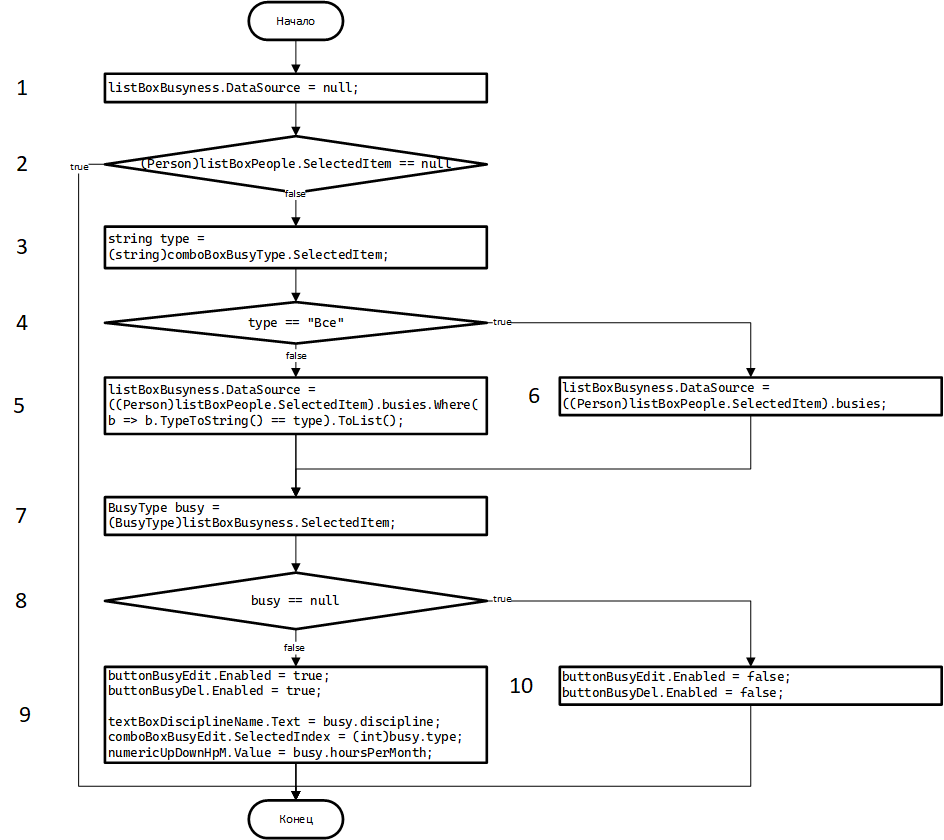


Рисунок 12 – Метод UpdateBusy

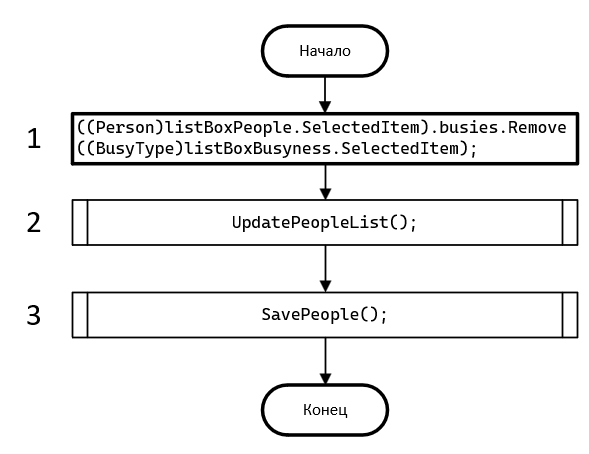


Рисунок 13 – Удаление занятости

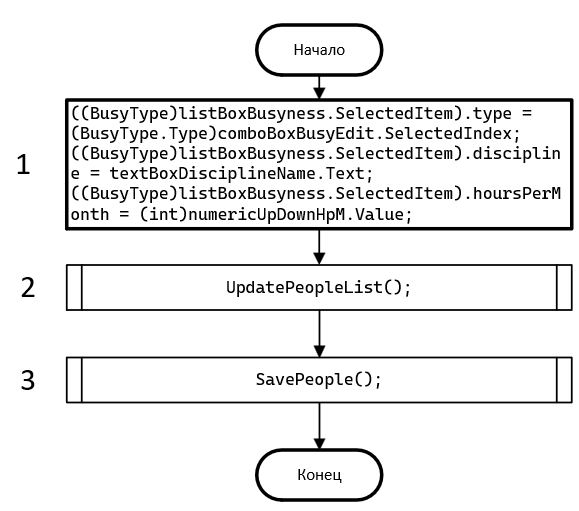


Рисунок 14 – Изменение занятости преподавателя

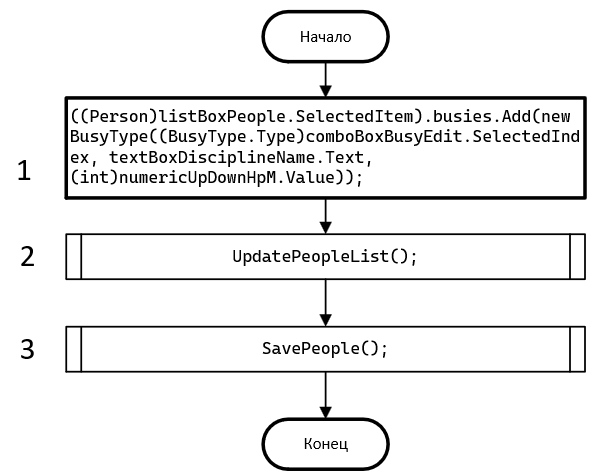


Рисунок 15 – Добавление занятости преподавателя

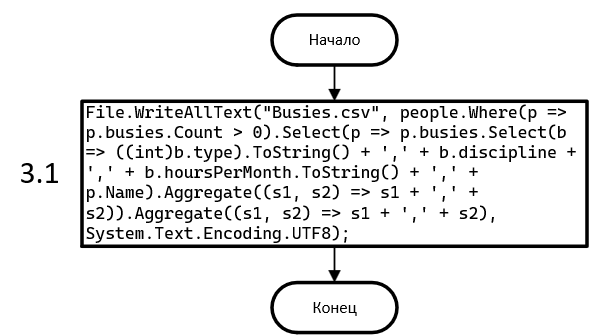


Рисунок 16 – Метод SaveBusy

1. **Подбор тестовых примеров**

В качестве тестовых данных созданы два преподавателя и два инженера со следующими занятостями:

Инженер Иванов И.И.:

Методическая работа, 20 часов

Геология, практика, 11 часов

Геология, лабораторная работа, 8 часов

Геология, научно-исследовательская работа, 5 часов

Преподаватель Петров А.Б.:

Организационная работа, 35 часов

Методическая работа, 30 часов

Преподаватель Мюслин Е.Н.:

Программирование, практика. 9 часов

Астрономия, лабораторная работа. 27 часов

ОБЖ, лекция. 18 часов

Программирование, лекция. 25 часов

Для хранения данных *csv*-фалы были заполнены следующим содержанием:

*People.csv*:

Иванов И.И.,f,Петров А.Б.,t,Мюслин Е.Н.,t,Андреев К.А.,f

*Busies.csv*:

2,-,20,Иванов И.И.,4,Геология,11,Иванов И.И.,5,Геология,8,Иванов И.И.,0,Геология,5,Иванов И.И.,1,-,35,Петров А.Б.,2,-,30,Петров А.Б.,4,Программирование,9,Мюслин Е.Н.,5,Астрономия,27,Мюслин Е.Н.,3,ОБЖ,18,Мюслин Е.Н.,3,Программирование,25,Мюслин Е.Н.

1. **Листинг (код) составленного программного обеспечения**

**Файл *MainForm.cs***

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Windows.Forms;

namespace LINQ

{

public partial class MainForm : Form

{

private List<Person> people = new List<Person>();

public MainForm()

{

InitializeComponent();

string[] csvData = File.ReadAllText("People.csv", System.Text.Encoding.UTF8).Split(',');

for (int i = 0; i < csvData.Length; i += 2)

people.Add(new Person(csvData[i], csvData[i + 1] == "t"));

csvData = File.ReadAllText("Busies.csv", System.Text.Encoding.UTF8).Split(',');

for (int i = 0; i < csvData.Length; i += 4)

people.Find(p => p.Name == csvData[i + 3]).busies.Add(new BusyType((BusyType.Type)int.Parse(csvData[i]), csvData[i + 1], int.Parse(csvData[i + 2])));

List<string> cbBTypes = people.SelectMany(p => p.busies).Select(b => b.discipline).Distinct().Except(new string[] { "-" }).ToList();

cbBTypes.Insert(0, "Все");

comboBoxDiscipline.DataSource = cbBTypes;

UpdatePeopleList();

}

public void UpdatePeopleList()

{

listBoxPeople.DataSource = null;

string discipline = (string)comboBoxDiscipline.SelectedItem;

string type = (string)comboBoxType.SelectedItem;

if (discipline == "Все")

listBoxPeople.DataSource = people;

else if (type == "Все")

listBoxPeople.DataSource = people.Where(p => p.busies.Any(b => b.discipline == discipline)).ToList();

else

listBoxPeople.DataSource = people.Where(p => p.busies.Any(b => b.discipline == discipline && b.TypeToString() == type)).ToList();

}

private void listBoxPeople\_SelectedIndexChanged(object sender, System.EventArgs e)

{

listBoxBusyness.DataSource = null;

Person person = (Person)listBoxPeople.SelectedItem;

if (person == null)

{

buttonPersonDell.Enabled = false;

buttonPersonEdit.Enabled = false;

buttonBusyEdit.Enabled = false;

buttonBusyDel.Enabled = false;

buttonBusyAdd.Enabled = false;

return;

}

buttonPersonDell.Enabled = true;

buttonPersonEdit.Enabled = true;

buttonBusyEdit.Enabled = true;

buttonBusyDel.Enabled = true;

buttonBusyAdd.Enabled = true;

textBoxName.Text = person.Name;

if (person.isTeacher)

radioButtonTeacher.Checked = true;

else

radioButtonEngineer.Checked = true;

comboBoxBusyType.DataSource = null;

comboBoxBusyType.DataSource = (new string[] { "Все" }).Concat(person.busies.Select(b => b.TypeToString())).Distinct().ToList();

listBoxBusyness.DataSource = ((Person)listBoxPeople.SelectedItem).busies;

UpdateBusyList();

}

private void comboBoxDiscipline\_SelectedIndexChanged(object sender, System.EventArgs e)

{

if (comboBoxDiscipline.SelectedItem == null)

return;

comboBoxType.Enabled = ((string)comboBoxDiscipline.SelectedItem != "Все");

comboBoxType.SelectedIndex = 0;

UpdatePeopleList();

}

private void comboBoxType\_SelectedIndexChanged(object sender, System.EventArgs e)

{

if (comboBoxType.SelectedItem == null)

return;

UpdatePeopleList();

}

private void comboBoxBusyType\_SelectedIndexChanged(object sender, System.EventArgs e)

{

if (listBoxPeople.SelectedItem == null)

return;

UpdateBusyList();

}

private void listBoxBusyness\_SelectedIndexChanged(object sender, System.EventArgs e)

{

UpdateBusyList();

}

private void UpdateBusyList()

{

listBoxBusyness.DataSource = null;

if ((Person)listBoxPeople.SelectedItem == null)

return;

string type = (string)comboBoxBusyType.SelectedItem;

if (type == "Все")

listBoxBusyness.DataSource = ((Person)listBoxPeople.SelectedItem).busies;

else

listBoxBusyness.DataSource = ((Person)listBoxPeople.SelectedItem).busies.Where(b => b.TypeToString() == type).ToList();

BusyType busy = (BusyType)listBoxBusyness.SelectedItem;

if (busy == null)

{

buttonBusyEdit.Enabled = false;

buttonBusyDel.Enabled = false;

return;

}

buttonBusyEdit.Enabled = true;

buttonBusyDel.Enabled = true;

textBoxDisciplineName.Text = busy.discipline;

comboBoxBusyEdit.SelectedIndex = (int)busy.type;

numericUpDownHpM.Value = busy.hoursPerMonth;

}

private void SavePeople()

{

File.WriteAllText("People.csv", people.Select(p => p.Name + ',' + (p.isTeacher ? 't' : 'f')).Aggregate((s1, s2) => s1 + ',' + s2), System.Text.Encoding.UTF8);

}

private void buttonPersonDell\_Click(object sender, System.EventArgs e)

{

people.Remove((Person)listBoxPeople.SelectedItem);

UpdatePeopleList();

SavePeople();

}

private void buttonPersonEdit\_Click(object sender, System.EventArgs e)

{

((Person)listBoxPeople.SelectedItem).Name = textBoxName.Text;

((Person)listBoxPeople.SelectedItem).isTeacher = radioButtonTeacher.Checked;

UpdatePeopleList();

SavePeople();

}

private void buttonPersonAdd\_Click(object sender, System.EventArgs e)

{

people.Add(new Person(textBoxName.Text, radioButtonTeacher.Checked));

UpdatePeopleList();

SavePeople();

}

private void SaveBusy()

{

File.WriteAllText("Busies.csv", people.Where(p => p.busies.Count > 0).Select(p => p.busies.Select(b => ((int)b.type).ToString() + ',' + b.discipline + ',' + b.hoursPerMonth.ToString() + ',' + p.Name).Aggregate((s1, s2) => s1 + ',' + s2)).Aggregate((s1, s2) => s1 + ',' + s2), System.Text.Encoding.UTF8);

}

private void buttonBusyDel\_Click(object sender, System.EventArgs e)

{

((Person)listBoxPeople.SelectedItem).busies.Remove((BusyType)listBoxBusyness.SelectedItem);

UpdateBusyList();

SaveBusy();

}

private void buttonBusyEdit\_Click(object sender, System.EventArgs e)

{

((BusyType)listBoxBusyness.SelectedItem).type = (BusyType.Type)comboBoxBusyEdit.SelectedIndex;

((BusyType)listBoxBusyness.SelectedItem).discipline = textBoxDisciplineName.Text;

((BusyType)listBoxBusyness.SelectedItem).hoursPerMonth = (int)numericUpDownHpM.Value;

UpdateBusyList();

SaveBusy();

}

private void buttonBusyAdd\_Click(object sender, System.EventArgs e)

{

((Person)listBoxPeople.SelectedItem).busies.Add(new BusyType((BusyType.Type)comboBoxBusyEdit.SelectedIndex, textBoxDisciplineName.Text, (int)numericUpDownHpM.Value));

UpdateBusyList();

SaveBusy();

}

}

}

**Файл *Person.cs***

using System.Collections.Generic;

using System.Linq;

namespace LINQ

{

public class Person

{

public string Name { get; set; }

public bool isTeacher { get; set; }

public List<BusyType> busies = new List<BusyType>();

public Person(string name, bool isTeacher)

{

Name = name;

this.isTeacher = isTeacher;

}

public override string ToString()

{

return (isTeacher ? "Преподаватель " : "Инженер ") + Name + " Всего ч/м: " + busies.Sum(b => b.hoursPerMonth);

}

}

}

**Файл *BusyType.cs***

namespace LINQ

{

public class BusyType

{

public enum Type { ScienseWork, Management, Metodical, Lecture, Practice, LabWork }

public Type type { get; set; }

public string discipline { get; set; }

public int hoursPerMonth { get; set; }

public BusyType(Type type, string discipline, int hoursPerMonth)

{

this.type = type;

this.discipline = discipline;

this.hoursPerMonth = hoursPerMonth;

}

public string TypeToString()

{

if (type == Type.ScienseWork)

return "Научно-исследовательская работа";

else if (type == Type.Management)

return "Организационная работа";

else if (type == Type.Metodical)

return "Методическая работа";

else if (type == Type.Lecture)

return "Лекция";

else if (type == Type.Practice)

return "Практика";

else if (type == Type.LabWork)

return "Лабораторная работа";

return "NotFound";

}

public override string ToString()

{

return TypeToString() + ", " + discipline + ". " + hoursPerMonth + "ч/м";

}

}

}

1. **Графический пользовательский интерфейс программного обеспечения и его описание**

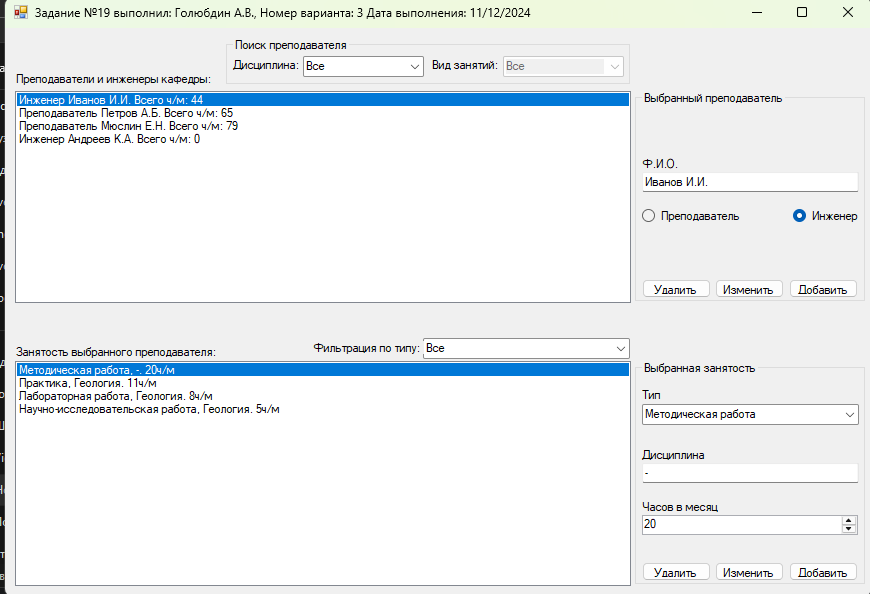


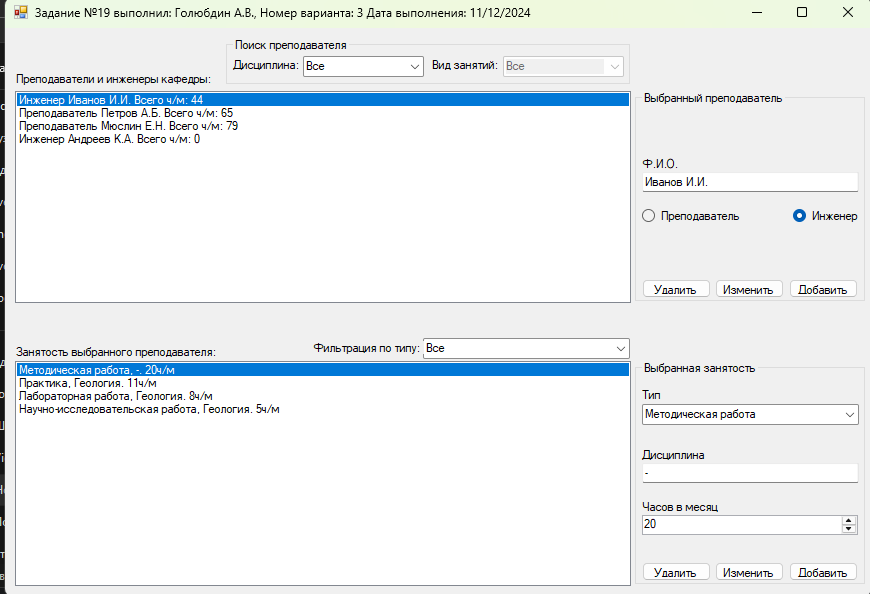
Рисунок 17 – Интерфейс главной формы

Сверху формы находится блок «Поиск преподавателя», где мы можем выбрать Дисциплину и Вид занятий для фильтрации преподавателей.

Далее находится список преподавателей, отфильтрованный по выбранным параметрам. Справа от него находится блок «Выбранный преподаватель», в элементы которого выводятся данные о преподавателе. При нажатии кнопки «Удалить» внизу этого блока, преподаватель удаляется из csv-файла. При нажатии кнопки «Изменить» внизу этого блока, изменения, внесенные в элементы блока, сохраняются в csv-файл. При нажатии кнопки «Добавить», в csv-файл добавляется новый преподаватель, данные которого берутся из элементов блока.

Далее находится список занятостей выбранного преподавателя, отфильтрованный по типу занятий, указанному справа над списком. Справа от него находится блок «Выбранная занятость», в элементы которого выводятся данные о занятости. При нажатии кнопки «Удалить» внизу этого блока, занятость удаляется из csv-файла. При нажатии кнопки «Изменить» внизу этого блока, изменения, внесенные в элементы блока, сохраняются в csv-файл. При нажатии кнопки «Добавить», в csv-файл добавляется новая занятость, данные которой берутся из элементов блока.

1. **Подтверждение соответствия графического пользовательского интерфейса требованиям к оформлению**

 Рисунок 18– Подтверждение соответствия графического пользовательского интерфейса требованиям к оформлению.

В верхнем списке мы можем видеть сводную информацию обо всех работниках кафедры, включая имя, должность и количество часов.

В нижнем списке с помощью выпадающего меню мы можем вывести информацию о занятости преподавателя научно-исследовательской, методической и организационной работой.

С помощью выпадающих списков вверху формы мы можем выдать информацию о преподавателях, ведущих указанный вид занятий по указанной дисциплине.

Таким образом, графический интерфейс программы соответствует требованиям к разработке.

1. **Расчёт тестовых примеров с использованием составленного программного обеспечения**

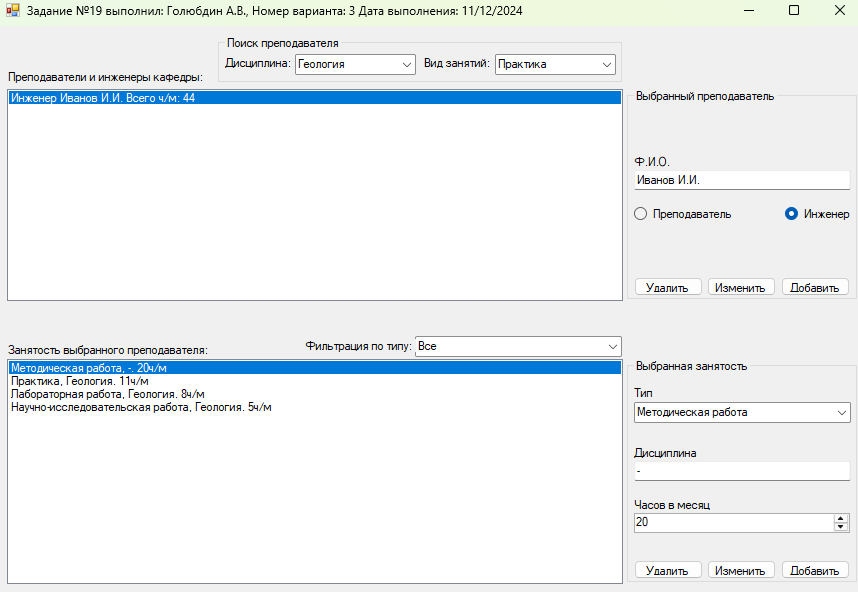


Рисунок 19 – Поиск преподавателя, ведущего практику по геологии

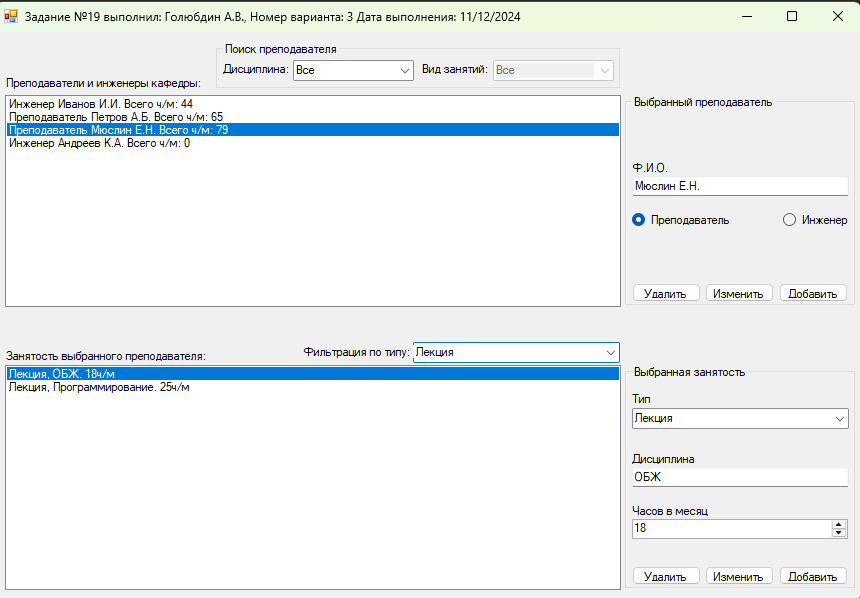
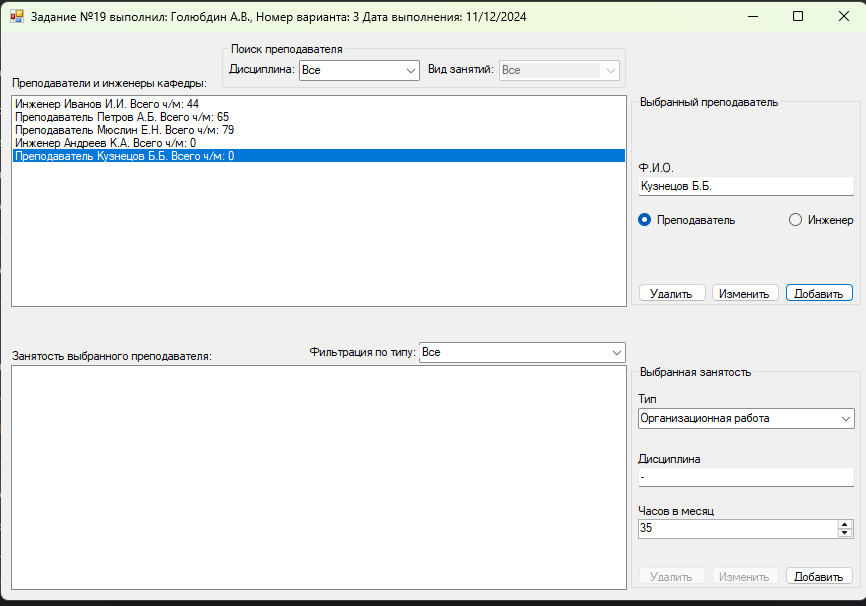


Рисунок 20 – Фильтрация занятости преподавателя по типу – «Лекция»

 Рисунок 21 – Добавление преподавателя с Ф.И.О. «Кузнецов Б.Б.»

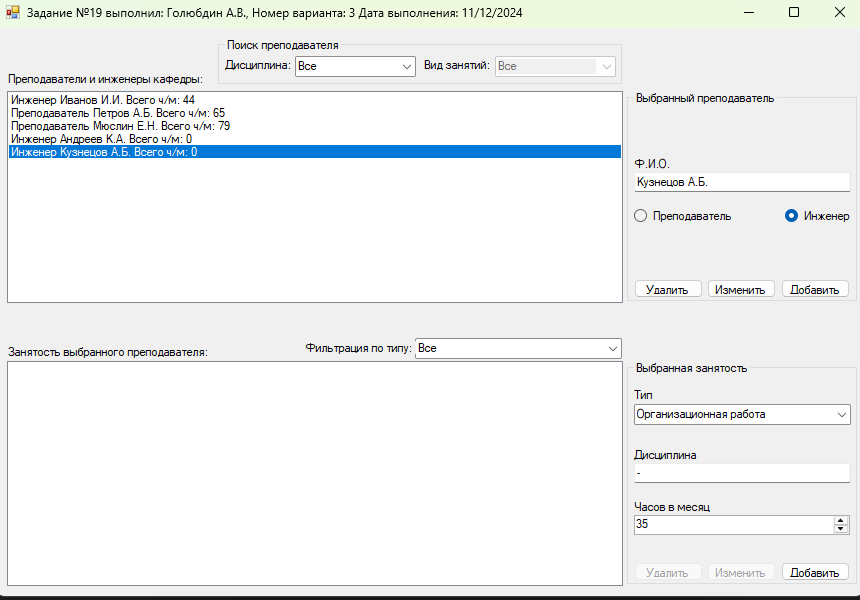


Рисунок 22 – Изменение должности и инициалов преподавателя с Ф.И.О. «Кузнецов Б.Б.»

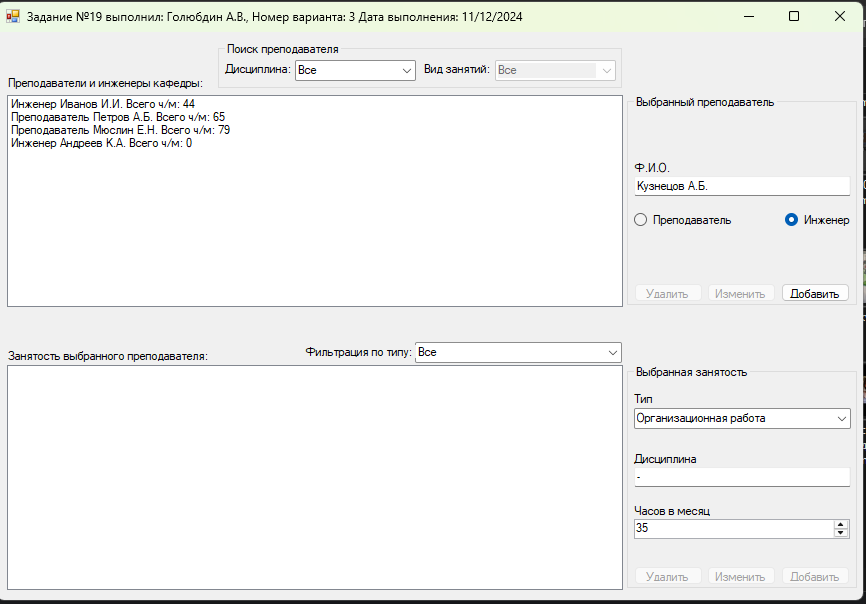


Рисунок 23 – Удаление инженера с Ф.И.О. «Кузнецов А.Б.»

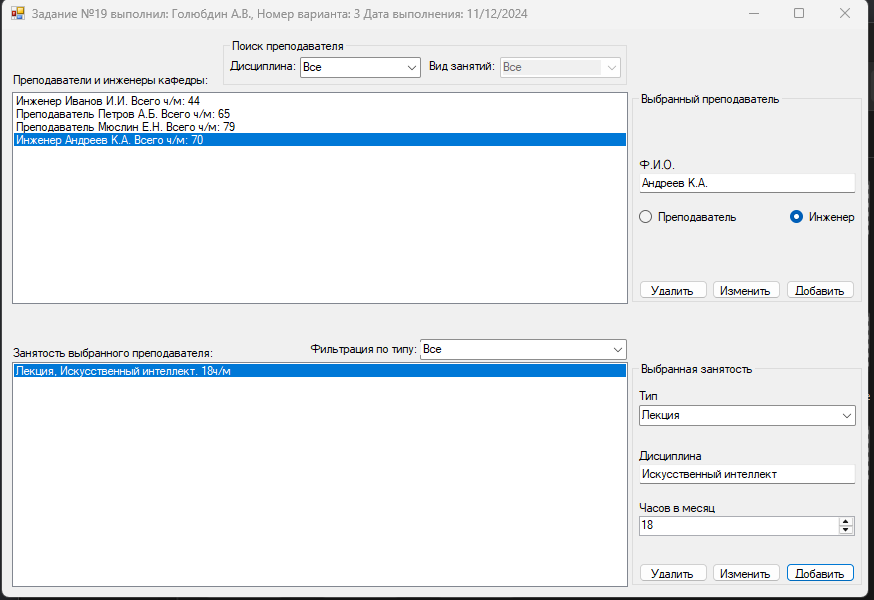


Рисунок 24 – Добавление Андрееву К.А. Лекции по Искусственному интеллекту на 18 часов в месяц.

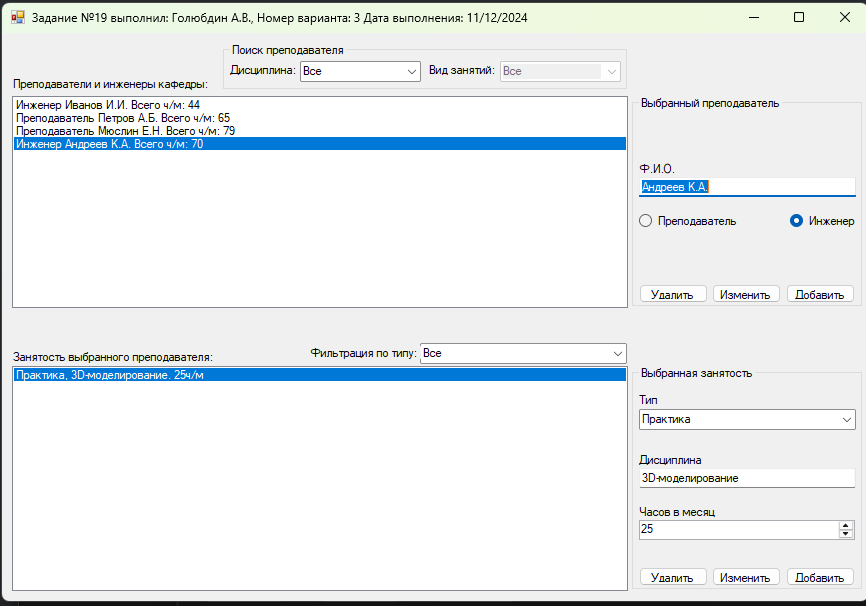


Рисунок 25 – Изменение всех данных добавленной лекции

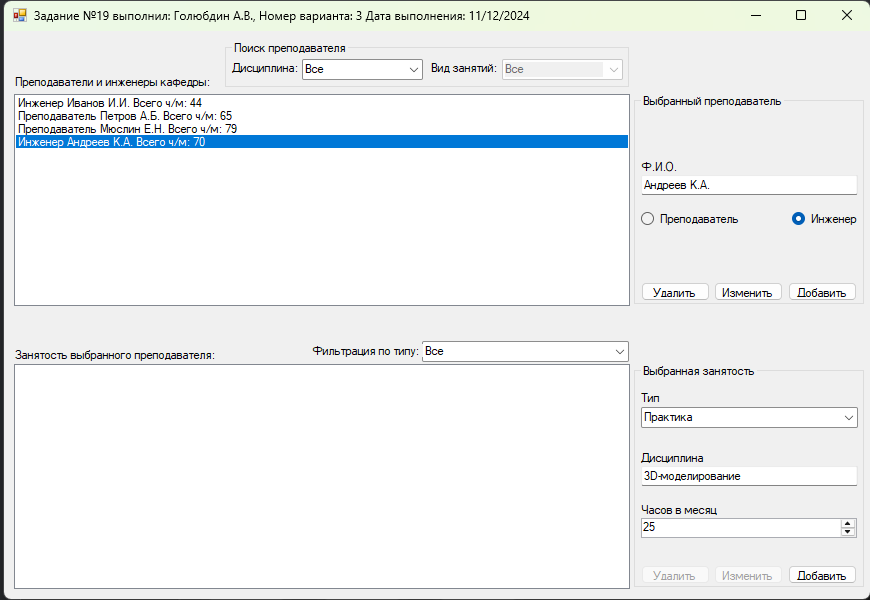


Рисунок 26 – Удаление добавленной занятости

1. **Формулировка вывода о проделанной работе**

Освоил навыки подготовки наборов данных по нечётко сформулированному описанию предметной области при учёте особенностей запросов, отклики на которые необходимо предоставлять заявленным (потенциальным) пользователям программного обеспечения.

Освоил навыки работы с библиотекой *LINQ* и применил их на практике в данной лабораторной работе.