|  |  |
| --- | --- |
|  | /storage/emulated/0/.polarisOffice5/polarisTemp/image1.png |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждениевысшего профессионального образования «Московский технологический университет»  МИРЭА | | |

Институт Информационных Технологий

Кафедра Корпоративных Информационных Систем

**ОТЧЕТ**

по Лабораторной Работе № 5

на тему

«Тип данных – структура»

по дисциплине

«Процедурное Программирование»

Выполнил студент группы ИКБО-07-17 Акжигитов Р. Р.

Принял асс. каф. КИС Алдобаева В. Н.

Выполнено «21» ноября 2017 г.

Зачтено «\_\_\_» \_\_\_\_\_\_\_\_\_ 2017 г.

Москва, 2017

**Задание**

Разработать приложение, позволяющее вносить данные в виде, определенном в задании. Данные должны быть представлены в памяти в виде коллекции структур. Обязательно должны быть реализованы функции: добавление элемента, удаление элемента, подсчет количества элементов. Дополнительно реализовать функционал, определенный в персональном варианте. Результат вывести в виде таблицы.

**Вариант №1**

Данные — сведения о студенте, включают: шифр, фамилию, имя, отчество, группу, факультет, направление подготовки.

1. Найти студентов с введенной фамилией.
2. Найти студентов на введенном факультете
3. Найти студентов в группе
4. Найти студентов по направлению подготовки, обучающихся НЕ на введенном факультете.

**Теоретическая часть**

**Структура** – это способ объединения данных в единую сущность. (Разнородные поля => единый элемент данных). В машинном коде структура практически не отличается от своих полей-переменных, представленных отдельно в памяти.

На практике часто используются массив структур, для реализации простой таблицы базы данных. Массив структур – сложная сущность, содержащая набор элементов с различной информацией.

С отдельной структурой можно работать как с единой переменной.

Сравнивать значения структур нельзя, так как компилятор не знает о смысле данных внутри структур и не может составить правильную операцию сравнения. Определить операцию сравнения структур можно самостоятельно:

**public static bool operator == (structure a, structure b) { return a.field1 == b.field2; }**

**public static bool operator != (structure a, structure b) { return a.field1 != b.field2; }**

Для работы с набором одинаковых структур, можно использовать массив записей, но более предпочтительным является использование списка, так как у него есть удобный интерфейс для добавления, удаления и поиска элементов, по сравнению с просто массивом.

**Person[] persons = new Person[123];**

**List<Person> persons = new List<Person>();**

Для нужно представить данные в таблице, где строки – единичные записи, столбцы – поля структуры данных, а ячейки – конкретные значения полей.

Для отображения таблицы существует два стандартных компонента: DataGridView (таблица строк) и ListView (возможно расширенное отображение элементов, произвольная отрисовка).

Каждый элемент списка ListView – это структура типа ListViewItem.

У него есть множество свойств:

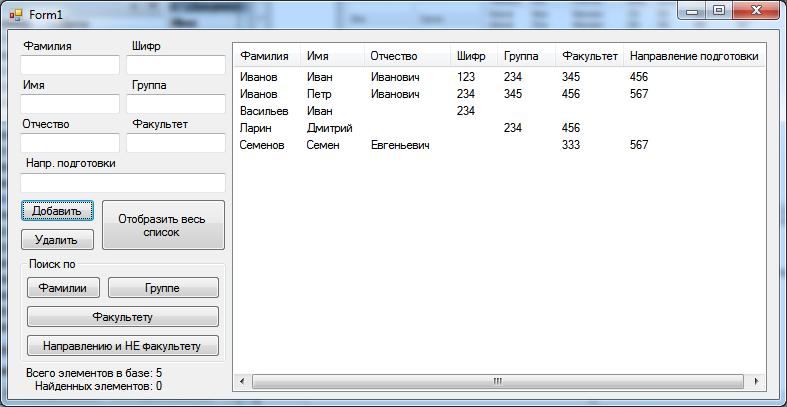
* **Text** – текст элемента, показываемый в первой колонке
* **Subitems** – набор дочерних элементов – ListViewItem, ListviewSubitem, хранящий текст и другие параметры ячеек, начиная со второй колонки
* **Tag** – произвольный объект (компонент сам никак не обрабатывает это поле).

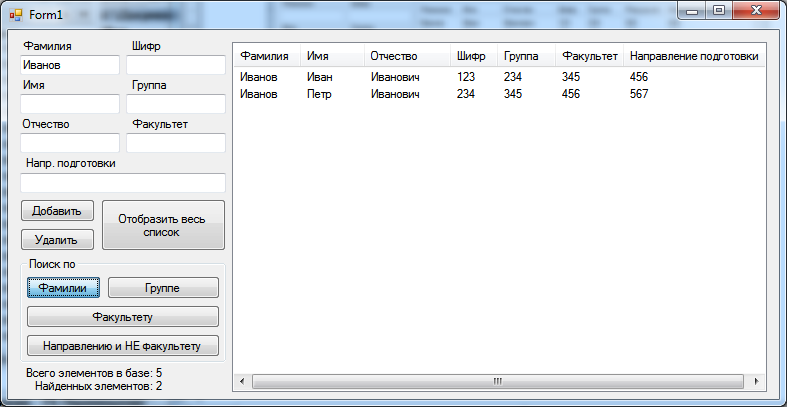
С помощью Tag мы можем связать строку в таблице и конкретный объект. Для того, чтобы при выделении определенной строки, можно было сослаться на объект (удалить его или изменить).

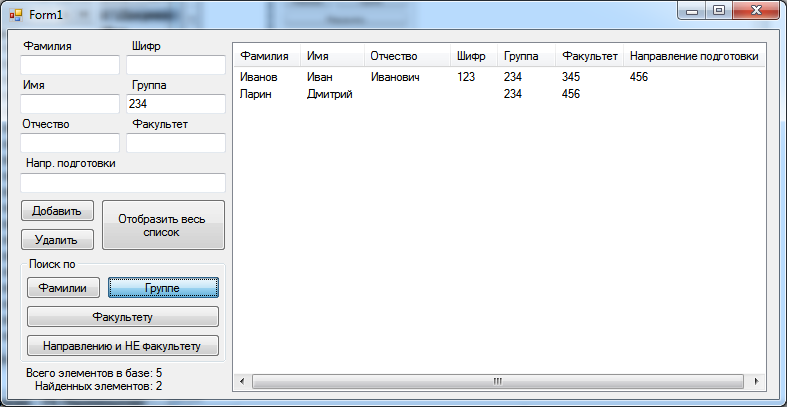
**Алгоритм решения задачи**

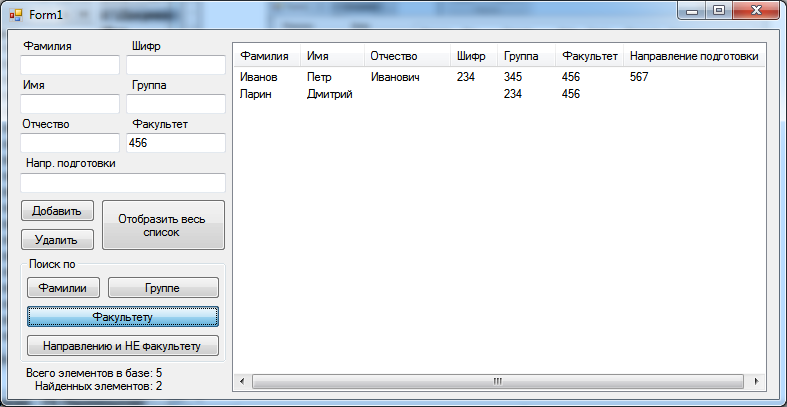
1. Создать форму с необходимыми компонентами (множество полей для заполнения таблицы, сама таблица).
2. Написать функции для создания таблицы в ListView (добавление элементов), добавление к общую базу, ресурс – введенные поля.
3. Написать функции обработки списка структур в соответствии с индивидуальным вариантом.

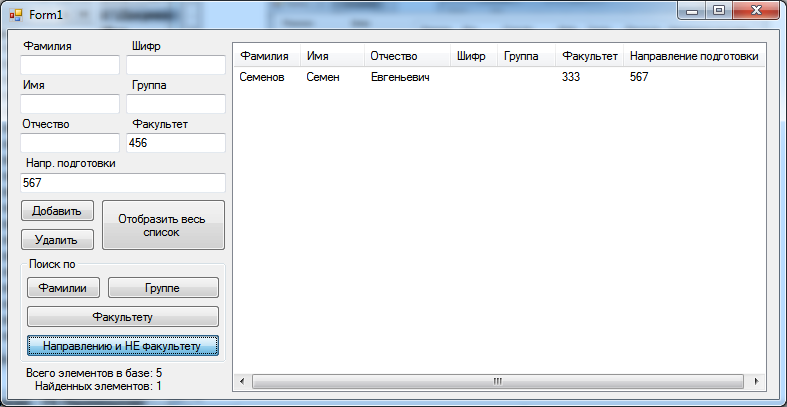
**Тестирование**











**Заключение**

В ходе выполнения работы был получен ценный опыт работы с структурами в C#, отображением таблиц на форме в компоненте ListView, а также обработка множества элементов (структур) списка.

**Исходный код**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace laba5

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

public struct Student

{

public string cipher;

public string name, lastName, parentsName;

public string faculty;

public string group;

public string studyDirection;

}

List<Student> database = new List<Student>();

List<Student> total = new List<Student>();

public void ListViewShow(Student obj, bool needClear = false)

{

if (needClear)

listView1.Items.Clear();

ListViewItem listLastName = new ListViewItem();

ListViewItem.ListViewSubItem subName = new ListViewItem.ListViewSubItem();

ListViewItem.ListViewSubItem subParentsName = new ListViewItem.ListViewSubItem();

ListViewItem.ListViewSubItem subCipher = new ListViewItem.ListViewSubItem();

ListViewItem.ListViewSubItem subGroup = new ListViewItem.ListViewSubItem();

ListViewItem.ListViewSubItem subFaculty = new ListViewItem.ListViewSubItem();

ListViewItem.ListViewSubItem subStudyDirection = new ListViewItem.ListViewSubItem();

listLastName.Text = obj.lastName;

listLastName.Tag = obj;

subName.Text = obj.name;

subParentsName.Text = obj.parentsName;

subCipher.Text = obj.cipher;

subGroup.Text = obj.group;

subFaculty.Text = obj.faculty;

subStudyDirection.Text = obj.studyDirection;

listLastName.SubItems.Add(subName);

listLastName.SubItems.Add(subParentsName);

listLastName.SubItems.Add(subCipher);

listLastName.SubItems.Add(subGroup);

listLastName.SubItems.Add(subFaculty);

listLastName.SubItems.Add(subStudyDirection);

listView1.Items.Add(listLastName);

}

public void ListViewShow(List<Student> database, bool needClear = false)

{

if (needClear)

listView1.Items.Clear();

foreach (var obj in database)

{

ListViewShow(obj);

}

}

public Student CreateStudentStruct()

{

Student newStudent;

newStudent.lastName = textBox1.Text;

newStudent.name = textBox2.Text;

newStudent.parentsName = textBox3.Text;

newStudent.cipher = textBox4.Text;

newStudent.group = textBox5.Text;

newStudent.faculty = textBox6.Text;

newStudent.studyDirection = textBox7.Text;

database.Add(newStudent);

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

textBox4.Text = "";

textBox5.Text = "";

textBox6.Text = "";

textBox7.Text = "";

return newStudent;

}

public void UpdateCounters(List<Student> db, int CountMode = 0)

{

if (CountMode == 0) label8.Text = "Всего элементов в базе: " + db.Count();

else if (CountMode == 1) label9.Text = "Найденных элементов: " + db.Count();

}

private void button1\_Click(object sender, EventArgs e)

{

CreateStudentStruct();

ListViewShow(database, true);

UpdateCounters(database);

total = database.ToList();

}

private void button2\_Click(object sender, EventArgs e)

{

total = database.Where(x => x.lastName == textBox1.Text).ToList();

ListViewShow(total, true);

UpdateCounters(total, 1);

}

private void button3\_Click(object sender, EventArgs e)

{

total = database.Where(x => x.faculty == textBox6.Text).ToList();

ListViewShow(total, true);

UpdateCounters(total, 1);

}

private void button4\_Click(object sender, EventArgs e)

{

total = database.Where(x => x.group == textBox5.Text).ToList();

ListViewShow(total, true);

UpdateCounters(total, 1);

}

private void button5\_Click(object sender, EventArgs e)

{

total = database.Where(x => x.studyDirection == textBox7.Text).Where(x => x.faculty != textBox6.Text).ToList();

ListViewShow(total, true);

UpdateCounters(total, 1);

}

private void button6\_Click(object sender, EventArgs e)

{

total = database.ToList();

ListViewShow(database.ToList(), true);

}

private void button7\_Click(object sender, EventArgs e)

{

foreach (ListViewItem item in listView1.SelectedItems)

{

database.Remove((Student)item.Tag);

total.Remove((Student)item.Tag);

}

ListViewShow(total, true);

UpdateCounters(total);

}

}

}