Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

Массивы, строки, коллекции, кортежи C#

тема

Вариант 14

Преподаватель \_\_\_\_\_\_\_\_\_\_\_ А.А. Чикизов

подпись, дата инициалы, фамилия

Студент КИ15–16Б, 031510065 \_\_\_\_\_\_\_\_\_\_\_ Т.В. Радионов

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2017

# Задание

Описать класс для работы с одномерным массивом:

* Создать параметризованный список структур;
* Параметры в открытый конструктор структуры передавать через кортеж (tuple);
* Произвести сортировку используя интерфейс ICompareble по разным полям;
* Перегрузить метод ToString() структуры, использовать его для вывода на экран элементов коллекции

Описать структуру с именем NOTE, содержащую следующие поля:

* фамилия, имя;
* номер телефона;
* дата рождения (массив из трех чисел).

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из восьми элементов типа NOTE (записи должны быть упорядочены по трем первым цифрам номера телефона);
* вывод на экран информации о человеке, чья фамилия введена с клавиатуры (если такого нет, вывести соответствующее сообщение).

# Исходный код

namespace WpfApp1

{

struct NOTE : IComparable<NOTE>

{

public string SurnameName { get; set; }

public string Phone { get; set; }

public string DOB { get; set; }

/// <summary>

/// Конструктор с кортежем

/// </summary>

/// <param name="note"></param>

public NOTE(Tuple<string, string, string> note)

{

SurnameName = note.Item1;

Phone = note.Item2;

DOB = note.Item3;

}

/// <summary>

/// Перегрузка метода ToString()

/// </summary>

/// <returns></returns>

public override string ToString()

{

return String.Format("Фамилия Имя: {0}; Телефон: {1}; Дата рождения: {2}", SurnameName, Phone, DOB);

}

/// <summary>

/// Реализация интерфейса IComparable<NOTE>

/// </summary>

/// <param name="obj"></param>

/// <returns></returns>

public int CompareTo(NOTE obj)

{

return Phone.CompareTo(obj.Phone);

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace WpfApp1

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

const int countNotes = 8;

List<NOTE> notes = new List<NOTE>(countNotes);

ICollectionView view;

int position = 0;

public MainWindow()

{

InitializeComponent();

}

/// <summary>

/// Событие загрузки окна

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

notes = new List<NOTE>(countNotes);

DG\_Notes.ItemsSource = notes;

DG\_Notes.Columns[0].Header = "Фамилия Имя";

DG\_Notes.Columns[1].Header = "Телефон";

DG\_Notes.Columns[2].Header = "Дата рождения";

}

/// <summary>

/// Нажатие на кнопку "Добавить"

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void B\_AddNote\_Click(object sender, RoutedEventArgs e)

{

if (position == countNotes)

{

MessageBox.Show("Достигнут лимит для записи (максимальное количество - " + countNotes + ")");

return;

}

notes.Add(new NOTE(new Tuple<string, string, string>(MTB\_SurnameName.Text, MTB\_Phone.Text, MTB\_DOB.Text)));

DG\_Notes.ItemsSource = notes;

DG\_Notes.Items.Refresh();

position++;

}

/// <summary>

/// Событие при вводе символа в бокс поиска по фамилии

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void TB\_FindBySurname\_TextChanged(object sender, TextChangedEventArgs e)

{

List<NOTE> filteredNotes = new List<NOTE>();

foreach (NOTE note in notes)

{

if (note.SurnameName.Contains(TB\_FindBySurname.Text))

filteredNotes.Add(note);

}

if (filteredNotes.Count == 0)

{

filteredNotes.Add(new NOTE(new Tuple<string, string, string>("Записей нет", "", "")));

}

view = CollectionViewSource.GetDefaultView(filteredNotes);

DG\_Notes.ItemsSource = view;

DG\_Notes.Columns[0].Header = "Фамилия Имя";

DG\_Notes.Columns[1].Header = "Телефон";

DG\_Notes.Columns[2].Header = "Дата рождения";

DG\_Notes.Items.Refresh();

}

/// <summary>

/// Сортировка по номеру телефона

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void B\_SortByPhone\_Click(object sender, RoutedEventArgs e)

{

notes.Sort();

DG\_Notes.Items.Refresh();

}

}

}

# Результы

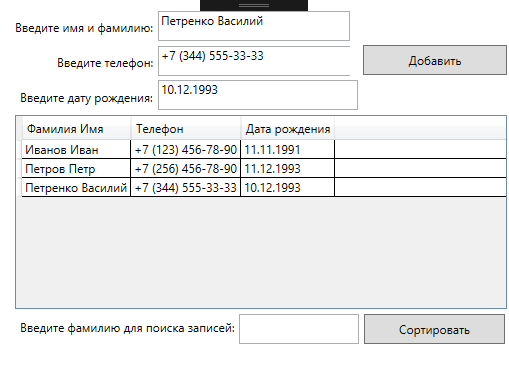


Рисунок 1 – Пример работы программы

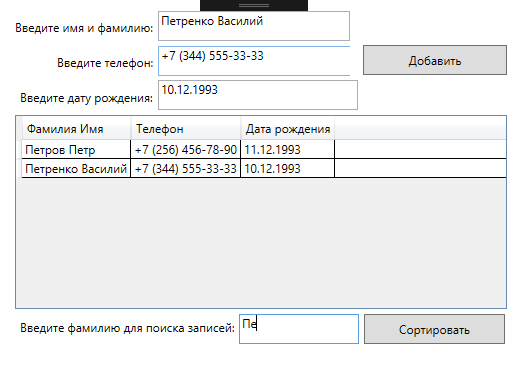


Рисунок 2 – Пример работы программы при поиске по фамилии

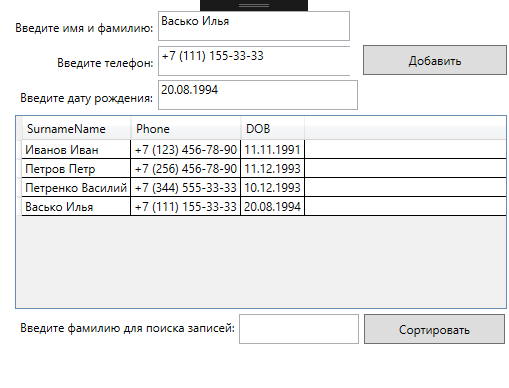


Рисунок 3 – Пример работы программы до сортировки по номеру

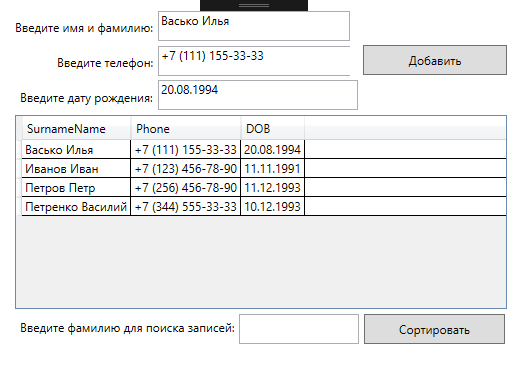


Рисунок 4 – Пример работы программы после сортировки по номеру