Федеральное агентство связи
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет
телекоммуникаций и информатики»

Лабораторная работа №11

Выполнил: студент 4 курса
ИВТ, гр. ИП-713
Михеев Н.А.
Проверил: ассистент кафедры
ПМиК
Агалаков А.А.

Новосибирск, 2020 г.

Цель

Объектно-ориентированный анализ, проектирование и реализация приложения под Windows «Телефонная книга».

Задание

Приложение должно обеспечивать пользователю:

- ввод, редактирование и сохранение имён абонентов городской
- телефонной сети и номеров их телефонов
- записи должны храниться и отображаться в отсортированном по
- именам порядке;
- поиск по имени;
- удаление записи;
- очистку книги.

Реализация и описание

- Для выполнения лабораторной работы был реализован класс UAbonentList в соответствии с заданием. Класс содержит единственное поле – contactsMap типа данных SortedDictionary с типами данных ключей и значений string. Данный тип данных является полным аналогом типа multimap из C++. Далее о каждом методе класса подробнее:
- public UAbonentList() – базовый конструктор класса, вызывается при старте программы. В нем инициализируется поле contactsMap и далее происходит чтение уже имеющихся контактов из фала contacts.txt, при наличии. Иначе создается данный файл;
- public bool Insert(string name, string number) – метод, принимающий в себя имя и номер абонента, идет валидация имени и телефона и если такой записи уже нет – добавляется в наш contactsMap;

- public bool Remove(string name) – метод, принимающий в себя имя абонента и при наличии такого в contactsMap – затирает запись данного абонента;
- public bool Edit(string name, string number) – метод, принимающий в себя имя абонента и номер, на который необходимо обновить запись. Сначала идет проверка существования данного контакта в книге, если такой имеется, то проверяется номер на правильность и заменяется;
- public bool Find(string name, string number) – метод, принимающий имя и телефон из программы. Далее если имя есть в книге – появляется messagebox с этой записью. Если поиск осуществлялся по телефону, то аналогично;
- public void Clear() – метод очищающий словарь контактов;
- public void Save() – вспомогательный метод, необходимый основной программе для сохранения имеющихся абонентов в файл;
- public void Load() – вспомогательный метод, необходимый основной прогрмме для загрузки абонентов из имеющегося файла.

Также были написаны тесты к каждому методу библиотеки.

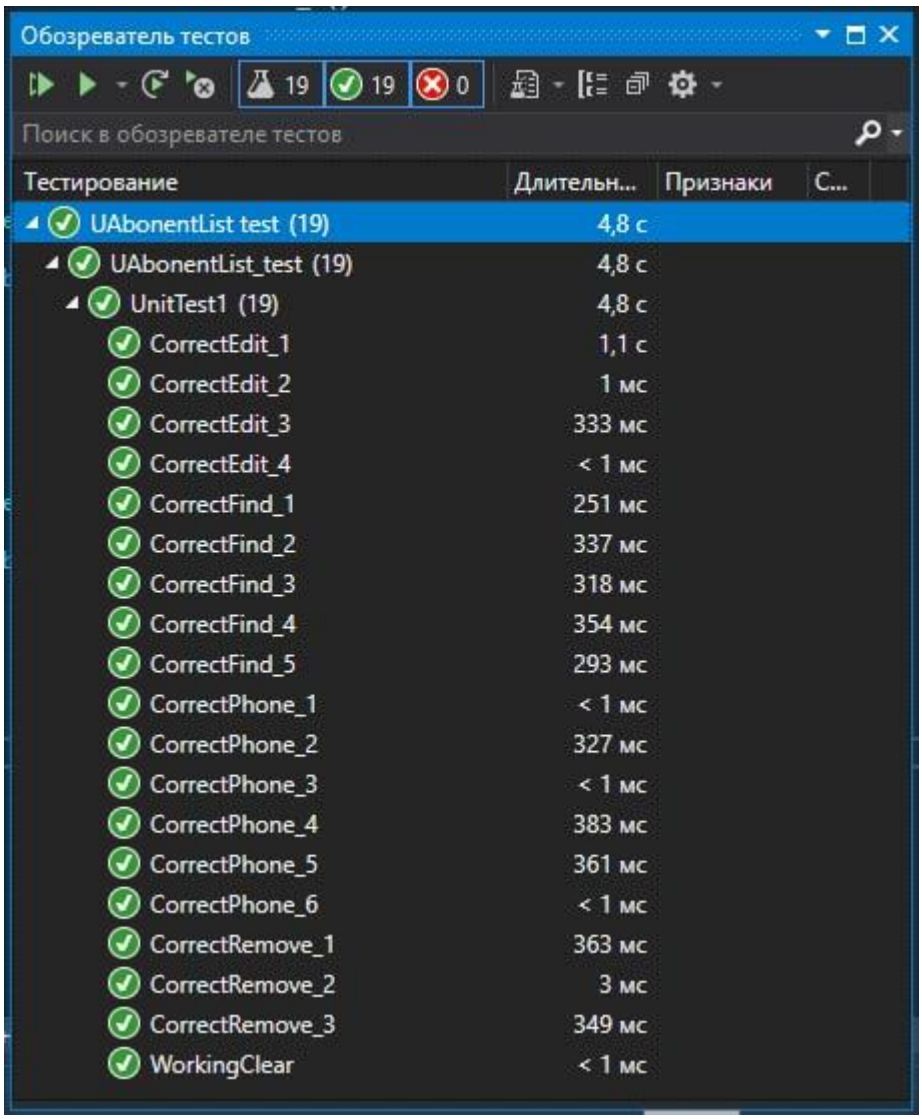

Рис.1 – демонстрация рабочих тестов

После реализации и тестирования библиотеки было создано графическое приложение под Windows с использованием инструментария Windows Forms.

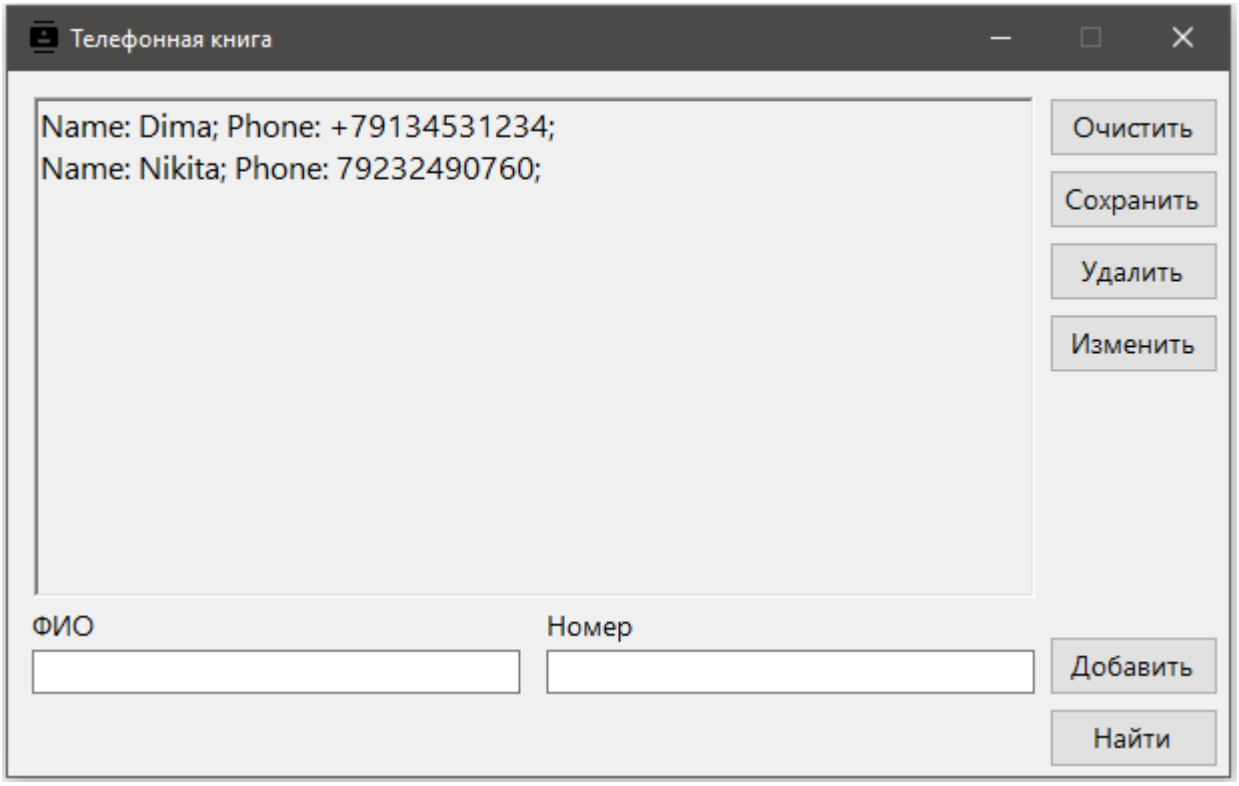

Рис.1 – демонстрация готового приложения.

Заключение

В ходе выполнения лабораторной работы был проведен объектно-ориентированный анализ, выполнено проектирование и реализация приложения под Windows «Телефонная книга» на языке C#. Было также успешно проведено модульное тестирование нашей библиотеки.

Код программы

UAbonentList.cs:

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.Net.Http;
using System.Security.Policy;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Lab11
{
    public class UAbonentList
    {
        public SortedDictionary<string, string> contactsMap;
```

```
        public UAbonentList()
        {
            contactsMap = new SortedDictionary<string, string>();
            if (File.Exists("contacts.txt"))
            {
                try
                {
                    using (StreamReader sr = new StreamReader("contacts.txt",
Encoding.Default))
                    {
                        string str;
                        while ((str = sr.ReadLine()) != null)
                        {
                            string[] entry;
                            string[] sep = { ";" };
                            entry = str.Split(sep,
StringSplitOptions.RemoveEmptyEntries);
                            contactsMap.Add(entry[0], entry[1]);
                        }
                    }
                }
                catch (Exception e)
                {
                    Debug.WriteLine("Error while reading file: " + e.ToString());
                }
            }
            else
            {
                File.Open("contacts.txt", FileMode.Create);
            }
        }

        public bool Insert(string name, string number)
        {
            Regex numReg = new Regex("\\+?\\d{11}");
            if(name.Length == 0 || !numReg.IsMatch(number))
            {
                MessageBox.Show("Wrong name or number.", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                return false;
            }
            if(contactsMap.ContainsKey(name))
            {
                MessageBox.Show("Abonent already exists.", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                return false;
            }
            contactsMap.Add(name, number);
            return true;
        }

        public bool Remove(string name)
        {
            if(!contactsMap.ContainsKey(name))
            {
                MessageBox.Show("No such abonent.", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                return false;
            }
            contactsMap.Remove(name);
            return true;
        }

        public bool Edit(string name, string number)
        {
```

```
            if(!contactsMap.ContainsKey(name))
            {
                MessageBox.Show("No such abonent.\nNothing to change", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                return false;
            }

            Regex numReg = new Regex("\\+?\\d{11}");
            if (!numReg.IsMatch(number))
            {
                MessageBox.Show("Wrong number.", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                return false;
            }
            contactsMap[name] = number;
            return true;
        }

        public bool Find(string name, string number)
        {
            if(name.Length != 0)
            {
                if(contactsMap.ContainsKey(name))
                {
                    MessageBox.Show("Name: " + name + "\nNumber: " + contactsMap[name],
"Success", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    return true;
                }
            }
            else if(number.Length != 0)
            {
                if (contactsMap.ContainsValue(number))
                {
                    MessageBox.Show("Name: " + contactsMap.FirstOrDefault(x => x.Value ==
number).Key + "\nNumber: " + number, "Success", MessageBoxButtons.OK,
MessageBoxIcon.Information);
                    return true;
                }
            }
            else if(name.Length == 0 && number.Length == 0)
            {
                MessageBox.Show("Empty search.", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                return false;
            }
            MessageBox.Show("No such abonent.", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            return false;
        }

        public void Clear()
        {
            contactsMap.Clear();
        }

        public void Save()
        {
            try
            {
                using(StreamWriter sw = new StreamWriter("contacts.txt"))
                {

                    foreach(KeyValuePair<String, String> kvp in contactsMap)
                    {
                        Console.WriteLine(kvp.Key + ";" + kvp.Value);
```

```
                        sw.WriteLine(kvp.Key + ";" + kvp.Value);
                    }
                }
                MessageBox.Show("File saved successfully.", "Successful",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            catch (Exception e)
            {
                Console.WriteLine("Error while opening file: " + e.ToString());
            }
        }

        public void Load()
        {
            if (File.Exists("contacts.txt"))
            {
                try
                {
                    using (StreamReader sr = new StreamReader("contacts.txt",
Encoding.Default))
                    {
                        string str;
                        while ((str = sr.ReadLine()) != null)
                        {
                            string[] entry;
                            string[] sep = { ";" };
                            entry = str.Split(sep,
StringSplitOptions.RemoveEmptyEntries);
                            contactsMap.Add(entry[0], entry[1]);
                        }
                    }
                }
                catch (Exception e)
                {
                    Debug.WriteLine("Error while reading file: " + e.ToString());
                }
            }
            else
            {
                File.Open("contacts.txt", FileMode.Create);
            }
        }

    }
}
```

Form1.cs:

```
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 Lab11
{
    public partial class Form1 : Form
    {
        public UAbonentList contacts;
        public Form1()
        {
            InitializeComponent();
            contacts = new UAbonentList();
```

```
            contacts.Load();
            UpdateTextBox();
        }

        private void UpdateTextBox()
        {
            richTextBox1.Clear();
            foreach(KeyValuePair<String, String> kvp in contacts.contactsMap)
            {
                richTextBox1.AppendText("Name: " + kvp.Key + "; " + "Phone: " + kvp.Value
+ ";\n");
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            contacts.Save();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            contacts.Remove(textBox1.Text);
            UpdateTextBox();
        }


        private void button1_Click(object sender, EventArgs e)
        {
            contacts.Clear();
            UpdateTextBox();
        }

        private void button4_Click(object sender, EventArgs e)
        {

            if (contacts.Edit(textBox1.Text, textBox2.Text))
            {
                textBox1.Clear();
                textBox2.Clear();
                UpdateTextBox();
            }
        }

        private void button6_Click(object sender, EventArgs e)
        {
            if(contacts.Insert(textBox1.Text, textBox2.Text))
            {
                textBox1.Clear();
                textBox2.Clear();
                UpdateTextBox();
            }
        }

        private void button7_Click(object sender, EventArgs e)
        {
            if(contacts.Find(textBox1.Text, textBox2.Text))
            {
                textBox1.Clear();
                textBox2.Clear();
            }
        }
    }
}
```

UnitTest1.cs:

```
using System;
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Lab11;

namespace UAbonentList_test
{
    [TestClass]
    public class UnitTest1
    {
        [TestMethod]
        public void CorrectPhone_1()
        {
            UAbonentList uAbonentList = new UAbonentList();
            Assert.IsTrue(uAbonentList.Insert("Vasya", "+79139504314"));
        }

        [TestMethod]
        public void CorrectPhone_2()
        {
            UAbonentList uAbonentList = new UAbonentList();
            Assert.IsFalse(uAbonentList.Insert("Vasya", "+791395043"));
        }

        [TestMethod]
        public void CorrectPhone_3()
        {
            UAbonentList uAbonentList = new UAbonentList();
            Assert.IsTrue(uAbonentList.Insert("Vasya", "79139504314"));
        }

        [TestMethod]
        public void CorrectPhone_4()
        {
            UAbonentList uAbonentList = new UAbonentList();
            Assert.IsFalse(uAbonentList.Insert("Vasya", ""));
        }

        [TestMethod]
        public void CorrectPhone_5()
        {
            UAbonentList uAbonentList = new UAbonentList();
            Assert.IsFalse(uAbonentList.Insert("", "79139504314"));
        }

        [TestMethod]
        public void CorrectPhone_6()
        {
            UAbonentList uAbonentList = new UAbonentList();
            Assert.IsTrue(uAbonentList.Insert("Vasya", "-79139504314"));
        }

        [TestMethod]
        public void CorrectRemove_1()
        {
            UAbonentList uAbonentList = new UAbonentList();
            uAbonentList.Insert("Vasya", "79139504314");
            Assert.IsFalse(uAbonentList.Remove(""));
        }

        [TestMethod]
        public void CorrectRemove_2()
        {
            UAbonentList uAbonentList = new UAbonentList();
            uAbonentList.Insert("Vasya", "79139504314");
```

```
    Assert.IsTrue(uAbonentList.Remove("Vasya"));
}

[TestMethod]
public void CorrectRemove_3()
{
    UAbonentList uAbonentList = new UAbonentList();
    uAbonentList.Insert("Vasya", "79139504314");
    Assert.IsFalse(uAbonentList.Remove("79139504314"));
}

[TestMethod]
public void CorrectEdit_1()
{
    UAbonentList uAbonentList = new UAbonentList();
    uAbonentList.Insert("Vasya", "79139504314");
    Assert.IsFalse(uAbonentList.Edit("Vasya", "123"));
}

[TestMethod]
public void CorrectEdit_2()
{
    UAbonentList uAbonentList = new UAbonentList();
    uAbonentList.Insert("Vasya", "79139504314");
    Assert.IsTrue(uAbonentList.Edit("Vasya", "+79139504314"));
}

[TestMethod]
public void CorrectEdit_3()
{
    UAbonentList uAbonentList = new UAbonentList();
    uAbonentList.Insert("Vasya", "79139504314");
    Assert.IsFalse(uAbonentList.Edit("V", "79139504314"));
}

[TestMethod]
public void CorrectEdit_4()
{
    UAbonentList uAbonentList = new UAbonentList();
    uAbonentList.Insert("Vasya", "79139504314");
    Assert.IsTrue(uAbonentList.Edit("Vasya", "+79131231234"));
}

[TestMethod]
public void CorrectFind_1()
{
    UAbonentList uAbonentList = new UAbonentList();
    uAbonentList.Insert("Vasya", "79139504314");
    Assert.IsTrue(uAbonentList.Find("Vasya", "79131231234"));
}

[TestMethod]
public void CorrectFind_2()
{
    UAbonentList uAbonentList = new UAbonentList();
    uAbonentList.Insert("Vasya", "79139504314");
    Assert.IsTrue(uAbonentList.Find("Vasya", ""));
}

[TestMethod]
public void CorrectFind_3()
{
    UAbonentList uAbonentList = new UAbonentList();
    uAbonentList.Insert("Vasya", "79139504314");
    Assert.IsTrue(uAbonentList.Find("", "79139504314"));
```

```
        }

        [TestMethod]
        public void CorrectFind_4()
        {
            UAbonentList uAbonentList = new UAbonentList();
            uAbonentList.Insert("Vasya", "79139504314");
            Assert.IsFalse(uAbonentList.Find("Valya", "79131231234"));
        }

        [TestMethod]
        public void CorrectFind_5()
        {
            UAbonentList uAbonentList = new UAbonentList();
            uAbonentList.Insert("Vasya", "79139504314");
            Assert.IsFalse(uAbonentList.Find("", "7913123123"));
        }

        [TestMethod]
        public void WorkingClear()
        {
            UAbonentList uAbonentList = new UAbonentList();
            uAbonentList.Insert("Vasya", "79139504314");
            try
            {
                uAbonentList.Clear();
            }
            catch (Exception e)
            {
                Assert.Fail(e.ToString());
            }
        }
    }
}
```