Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

ОТЧЕТ

по лабораторной работе № 6

по дисциплине

«Новые технологии в программировании»

Студент гр. 586-1

\_\_\_\_\_\_\_\_\_\_\_ А. Ахметов

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Руководитель

доцент каф. КСУП

\_\_\_\_\_\_\_\_\_\_\_ А.Е. Горяинов

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Томск 2020

На рисунке 1.1 представлена структура файлов репозитория в ветке develop.

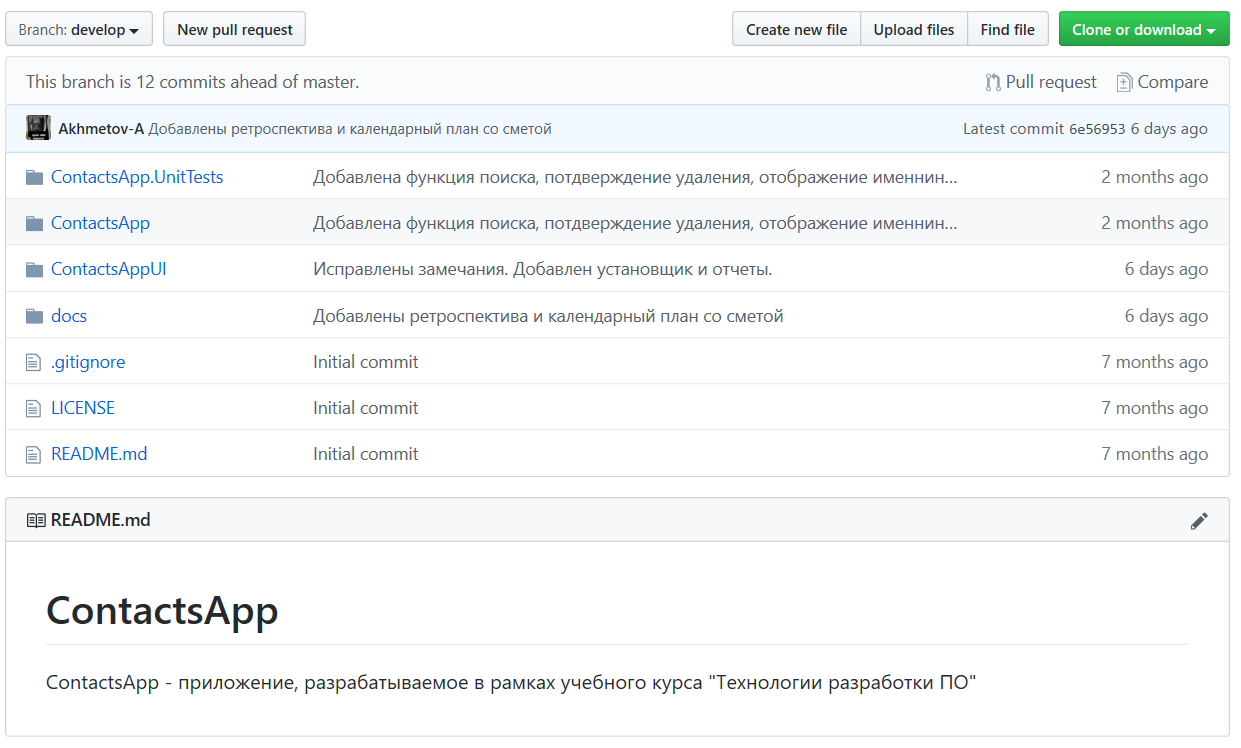


Рисунок 1.1 – Структура файлов репозитория в ветке develop.

На рисунке 1.2 представлена история коммитов ветки develop.

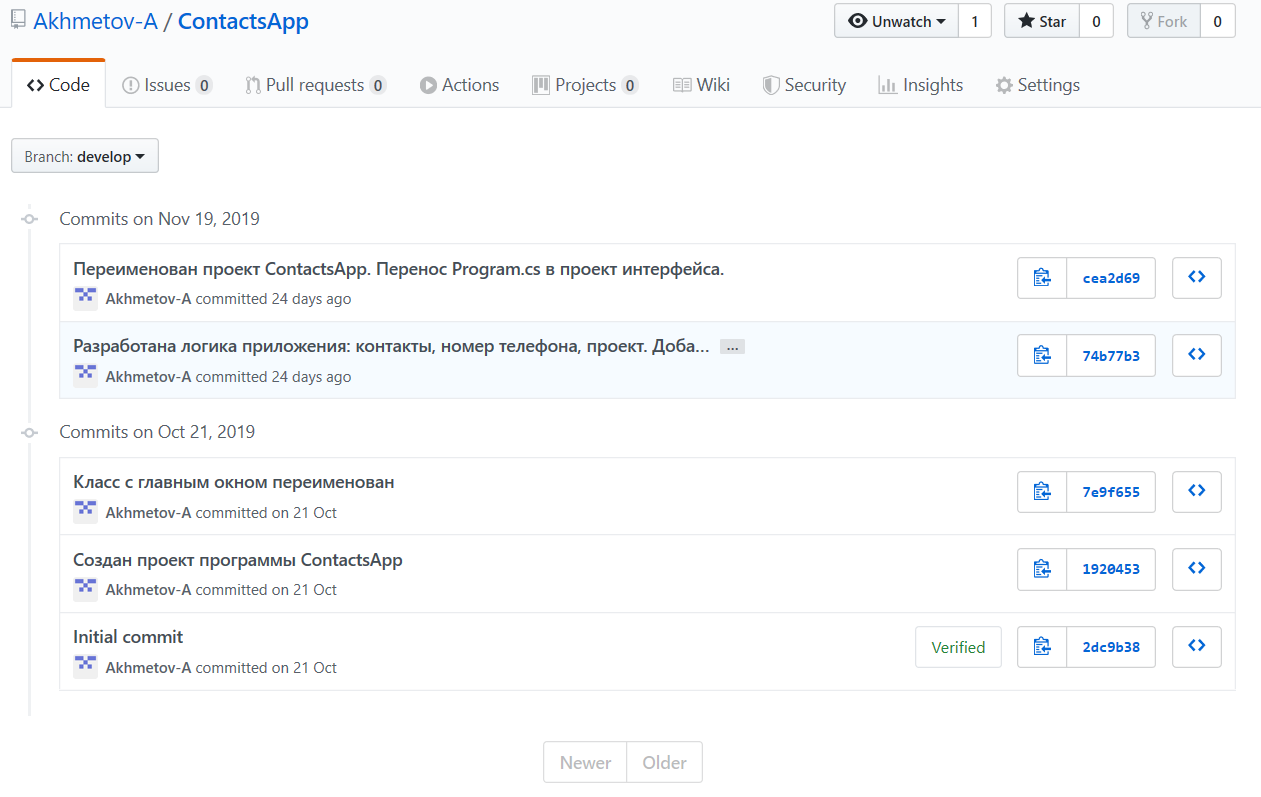


Рисунок 1.2 – История коммитов ветки develop.

На рисунке 2.1 представлена UML-диаграмма классов

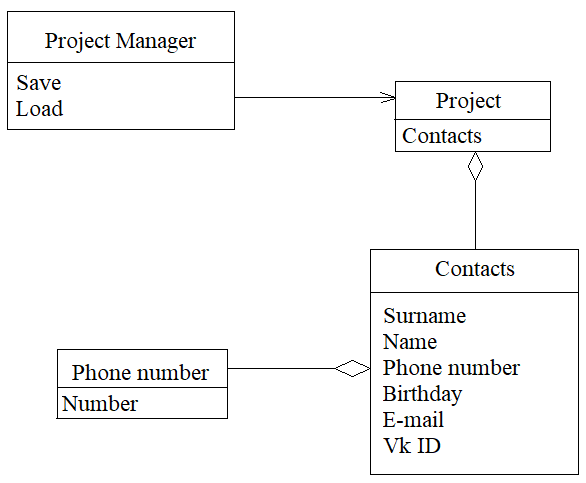


Рисунок 2.1 – UML-диаграмма классов

Ниже представлен код классов «ContactsApp»

Класс «Менеджер проекта»

namespace ContactsApp

{

/// <summary>

/// Класс "Менеджер проекта". Позволяет сохранять и загружать даннные.

/// </summary>

public static class ProjectManager

{

/// <summary>

/// Метод сохранения

/// </summary>

/// <param name="data">Переменная, которая хранит данные контактов и сохраняет их в текстовый файл</param>

/// <param name="filename">Именование файла</param>

public static void SaveToFile(Project data, string filename)

{

Newtonsoft.Json.JsonSerializer serializer = new Newtonsoft.Json.JsonSerializer();

//string documents = Environment.GetFolderPath(Environment.SpecialFolder.Personal);

using (StreamWriter sw = new StreamWriter(filename))

using(Newtonsoft.Json.JsonWriter writer = new Newtonsoft.Json.JsonTextWriter(sw))

{

serializer.Serialize(writer, data);

}

}

/// <summary>

/// Метод загрузки

/// </summary>

/// <param name="filename">Именование файла</param>

public static Project Load(string filename)

{

Project project = null;

Newtonsoft.Json.JsonSerializer serializer = new Newtonsoft.Json.JsonSerializer();

//string documents = Environment.GetFolderPath(Environment.SpecialFolder.Personal);

using (StreamReader sr = new StreamReader(filename))

using(Newtonsoft.Json.JsonReader reader = new Newtonsoft.Json.JsonTextReader(sr))

{

project = (Project)serializer.Deserialize<Project>(reader);

}

return project;

}

}

}

Класс «Проект»

namespace ContactsApp

{

/// <summary>

/// Класс "Проект". Представляет собой список контактов.

/// </summary>

public class Project

{

/// <summary>

/// Объявление списка.

/// </summary>

public List<Contact> Contacts = new List<Contact>();

public List<Contact> Sort(List<Contact> Contacts)

{

Contacts.Sort(delegate (Contact x, Contact y)

{

return x.Surname.CompareTo(y.Surname);

});

return Contacts;

}

public static Project Sort(Project project, string substring)

{

Project sortedProject = new Project();

for (int i = 0; i < project.Contacts.Count; i++)

{

if (project.Contacts[i].Surname.Contains(substring) ||

project.Contacts[i].Name.Contains(substring))

{

sortedProject.Contacts.Add(project.Contacts[i]);

}

}

if (sortedProject.Contacts.Count == 0)

{

sortedProject = null;

return sortedProject;

}

return sortedProject;

}

public static Project Birthday(Project data, DateTime today)

{

Project returnProject = new Project();

for (int i = 0; i < data.Contacts.Count; i++)

{

if ((data.Contacts[i].BirthDay.Day == today.Day) && (data.Contacts[i].BirthDay.Month == today.Month))

{

returnProject.Contacts.Add(data.Contacts[i]);

}

}

return returnProject;

}

}

}

Класс «Контакт»

namespace ContactsApp

{

/// <summary>

/// Класс "Контакт". Содержит фамилию, имя, дату рождения, e-mail, id вконтакте.

/// </summary>

public class Contact

{

/// <summary>

/// Поле "Фамилия".

/// </summary>

private string \_surname;

/// <summary>

/// Поле "Имя".

/// </summary>

private string \_name;

/// <summary>

/// Поле "День рождение".

/// </summary>

private DateTime \_birthDay;

/// <summary>

/// Поле "Электронная почта".

/// </summary>

private string \_email;

/// <summary>

/// Поле "ID Вконтакте".

/// </summary>

private string \_idvk;

/// <summary>

/// Свойство поля "Фамилия". Содержит не более 50 символов.

/// </summary>

public string Surname

{

get { return \_surname; }

set

{

if (value.Length > 50)

{

throw new ArgumentException("Длина фамилии не должна быть больше 50 символов");

}

else

\_surname = value;

}

}

/// <summary>

/// Свойство поля "Имя". Содержит не более 50 символов.

/// </summary>

public string Name

{

get { return \_name; }

set

{

if (value.Length > 50)

{

throw new ArgumentException("Длина имени не должна быть больше 50 символов");

}

else

\_name = value;

}

}

/// <summary>

/// Свойство поля "День рождения". Содержит информацию о дне, месяце, годе рождения владельца контакта.

/// </summary>

public DateTime BirthDay

{

get { return \_birthDay; }

set

{

if (value>DateTime.Today || value.Year<1900)

{

throw new ArgumentException("Дата рождения не может быть раньше 1900 года и позже текущего");

}

else

\_birthDay = value;

}

}

/// <summary>

/// Поле "Номер телефона". Реализация представлена в отдельном файле.

/// </summary>

public PhoneNumber Phone { get; set; } = new PhoneNumber();

/// <summary>

/// Свойство поля "Электронная почта". Содержит не более 50 символов.

/// </summary>

public string Email

{

get { return \_email; }

set

{

if (value.Length > 50)

{

throw new ArgumentException("Длина e-mail не должна быть больше 50 символов");

}

else

\_email = value;

}

}

/// <summary>

/// Свойство поля "ID Вконтакте". Содержит не более 15 символов.

/// </summary>

public string IDvk

{

get { return \_idvk; }

set

{

if (value.Length > 15)

{

throw new ArgumentException("Длина id не должна быть больше 15 символов");

}

else

\_idvk = value;

}

}

}

}

Класс «Номер телефона»

namespace ContactsApp

{

/// <summary>

/// Отдельный класс "Номер телефона".

/// </summary>

public class PhoneNumber

{

/// <summary>

/// Поле "Номер".

/// </summary>

private long \_number;

/// <summary>

/// Свойство поля "Номер". Начинается с 7 и состоит из 11 цифр.

/// </summary>

public long Number

{

get { return \_number; }

set

{

if (value < 70000000000 || value > 79999999999)

{

throw new ArgumentException("Номер телефона должен состоять из 11 цифр и начинаться с 7");

}

else

\_number = value;

}

}

}

}

На рисунке 2.2 представлен репозиторий текущей ветки на GitHub

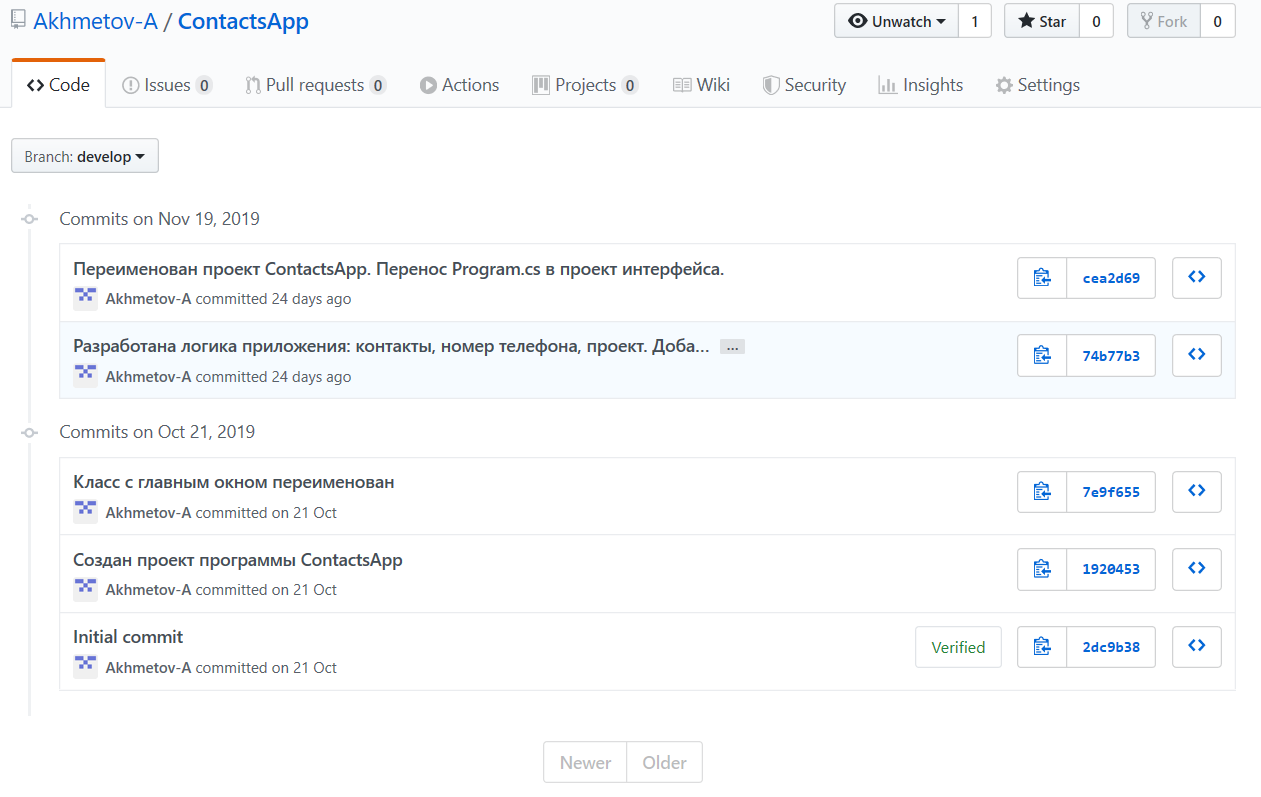


Рисунок 2.2 – репозиторий текущей ветки на GitHub

После запуска приложения перед пользователем появляется главное окно (рис. 3.1). Двухколоночная верстка главного окна содержит список всех контактов в левой панели и отображает информацию о текущем выбранном контакте в правой панели. В списке контактов показаны фамилии контактов.

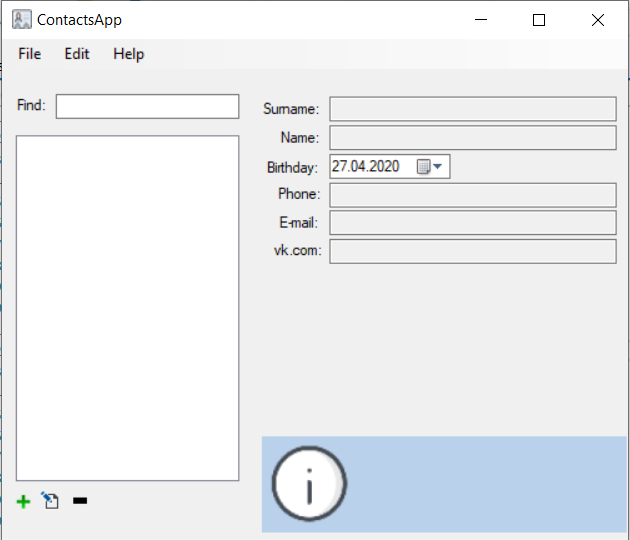


Рисунок 3.1 – Главное окно приложения ContactsApp

На панели со списком контактов внизу располагаются три кнопки: Add («Создать новый контакт»), Edit («Редактировать выбранный контакт»), Remove («Удалить выбранный контакт»).

Главное окно приложения не позволяет редактировать содержимое контакта – только его просмотр.

При нажатии на кнопку Add и Edit, или, при выборе соответствующих действий в верхней панели главного окна в пункте Edit, появляется окно создания / редактирования контакта в диалоговом режиме (рис. 3.2). Для нового контакта окно изначально не заполнено (рис. 3.3). Для редактирования уже существующего контакта все поля заполнены данными текущего контакта.

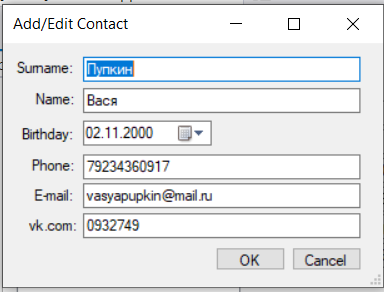


Рисунок 3.2 – Окно редактирования контакта в приложении ContactsApp

При редактировании текущего контакта, нажатие на кнопку ОК обновляет его данные в списк контактов.

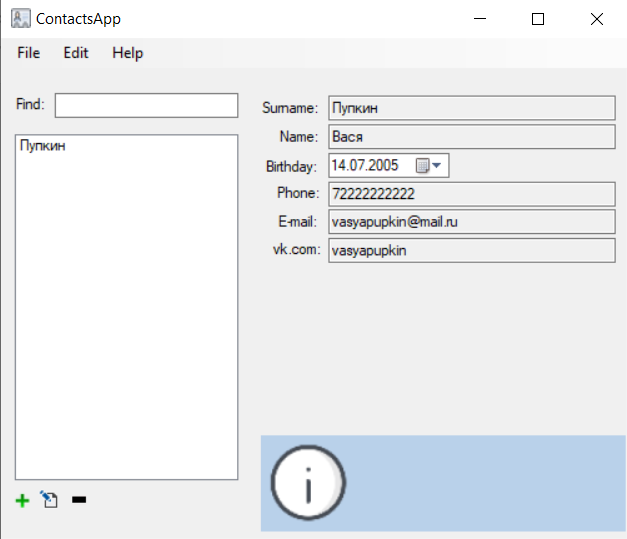


Рисунок 3.3 – Окно добавление контакта в приложении ContactsApp

При нажатии на кнопку ОК окно создания контакта закрывается, а в список контактов главного окна добавляется новый контакт.

Окно «About» (рис. 3.4) содержит информацию о названии приложения, его текущую версию, информацию о разработчике приложения, его почта, и ссылка на текущую версию приложения на GitHub.

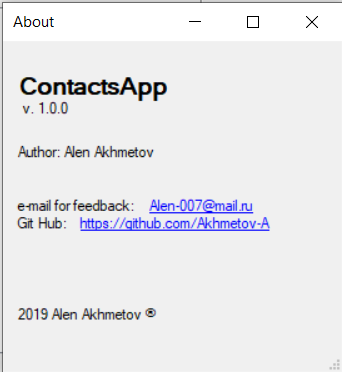


Рисунок 3.4 – Окно «About» о приложении ContactsApp

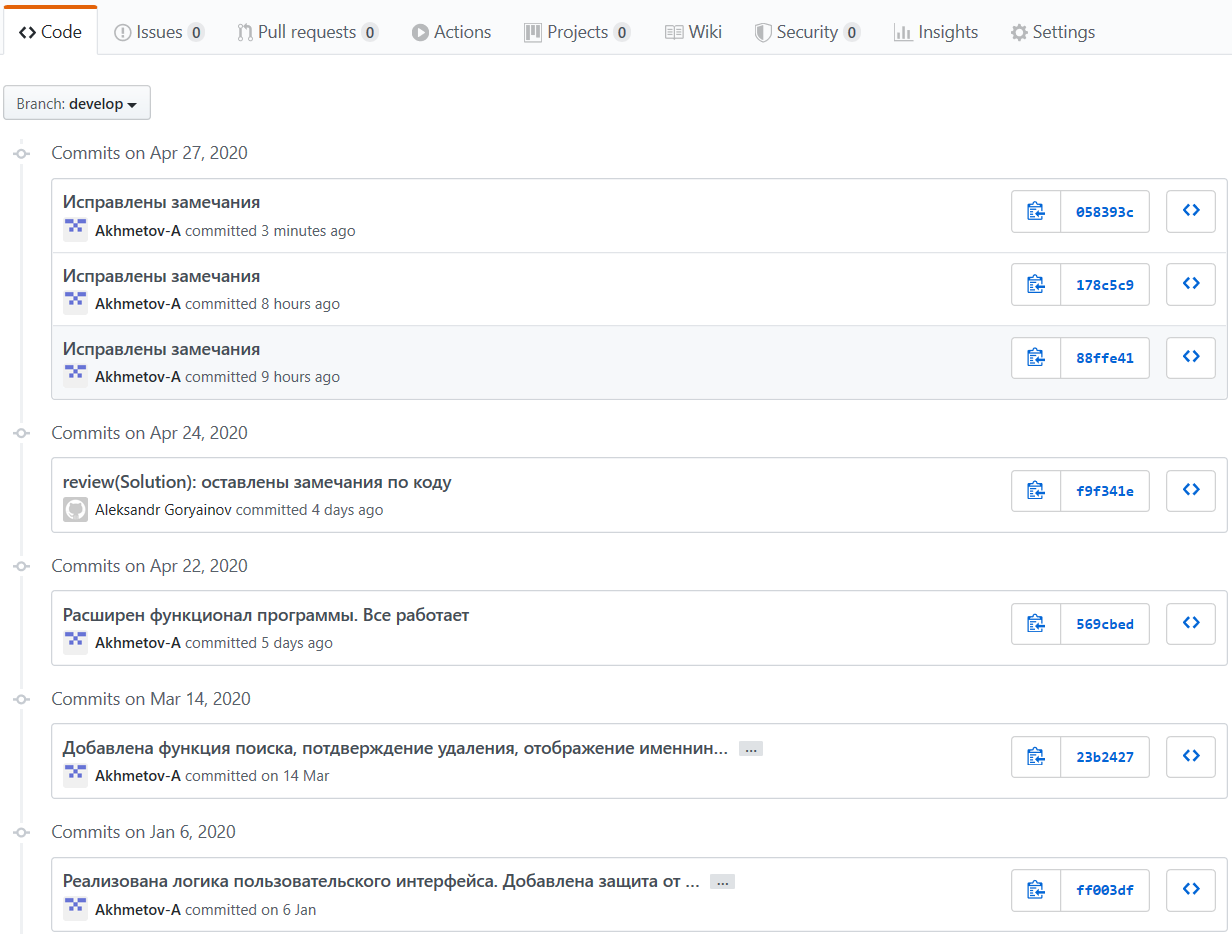


Рисунок 5 – Текущая история коммитов ветки develop

# Дерево цикломатической сложности методов и классов проекта логики

Проект ContactsApp (цикл. сложность проекта = 26)

1. Класс Contact (цикл. сложность класса = 20);

* Свойство Email (цикл. сложность = 3);
* Свойство Surname (цикл. сложность = 3);
* Свойство Name (цикл. сложность = 3);
* Свойство Birthday (цикл. сложность = 4);
* Свойство Id (цикл. сложность = 3);
* Свойство Phone (цикл. сложность = 4);

1. Класс PhoneNumber (цикл. сложность класса = 4);

* Свойство Number (цикл. сложность = 4);

1. Класс ProjectManager (цикл. сложность класса = 2);

* Метод SaveToFile (цикл. сложность = 1);
* Метод LoadFromFile (цикл. сложность = 1).

На рисунке 5.1 представлена актуальная UML-диаграмма классов

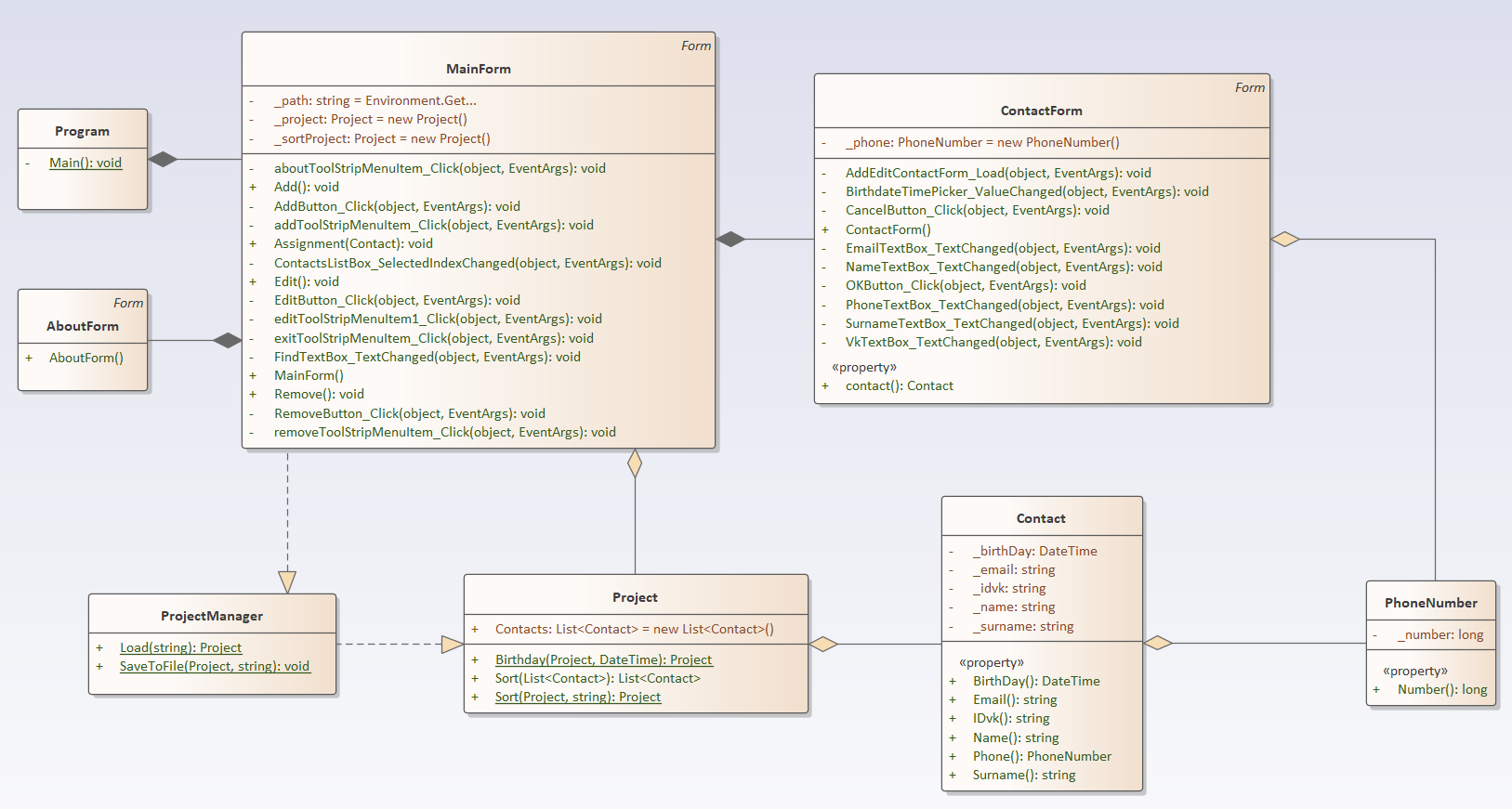


Рисунок 5.1 – UML-диаграмма классов

Для создания сборки установочного пакета необходимо прописать команды, которые представлены ниже.

#define MyAppName "ContactsApp"

#define MyAppVersion "1.0.0"

#Define MyAppPublisher "MyCompany"

#define MyAppURL "https://github.com/Akhmetov-A/ContactsApp"

#define MyAppExeName "ContactsApp.exe"

[Setup]

AppId={{46521D30-F550-44FE-A353-6BBC957792E9}

AppName={#MyAppName}

AppVersion={#MyAppVersion}

AppPublisher={#MyAppPublisher}

AppPublisherURL={#MyAppURL}

AppSupportURL={#MyAppURL}

AppUpdatesURL={#MyAppURL}

DefaultDirName={pf}\{#MyAppName}

DisableProgramGroupPage=yes

OutputBaseFilename=setup

Compression=lzma

SolidCompression=yes

[Languages]

Name: "english"; MessagesFile: "compiler:Default.isl"

[Tasks]

Name: "desktopicon"; Description: "{cm:CreateDesktopIcon}"; GroupDescription: "{cm:AdditionalIcons}"; Flags: unchecked

[Files]

Source: "..\Release\\*.exe"; DestDir: "{app}"; Flags: ignoreversion

Source: "..\Release\\*.dll"; DestDir: "{app}"; Flags: ignoreversion recursesubdirs createallsubdirs

[Icons]

Name: "{commonprograms}\{#MyAppName}"; Filename: "{app}\{#MyAppExeName}"

Name: "{commondesktop}\{#MyAppName}"; Filename: "{app}\{#MyAppExeName}"; Tasks: desktopicon

Установочный пакет содержит следующую перечень файлов: ContactsAppUI.exe, ContactsAppClasses.dll, NewtonsoftJson.dll.

На рисунке 5.2 представлена текущая история коммитов ветки develop.

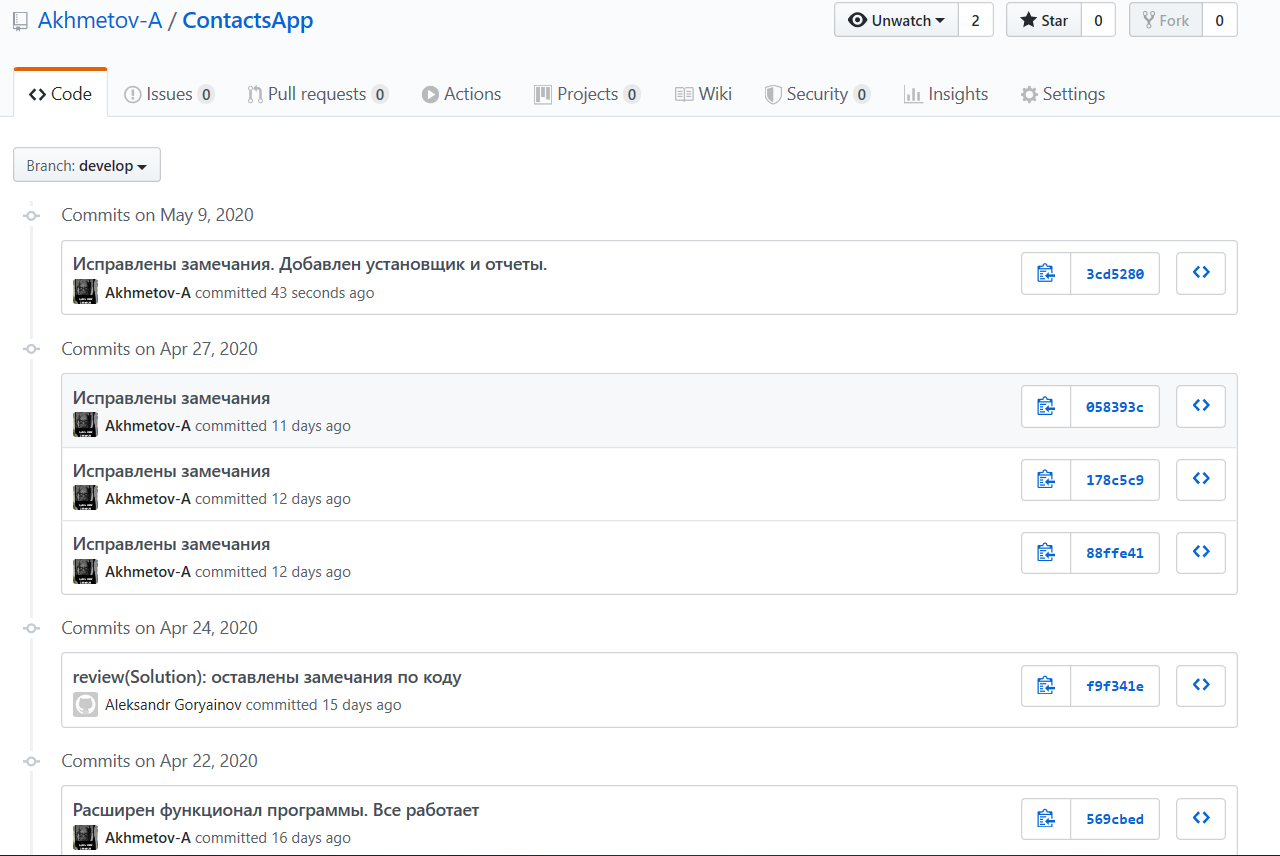


Рисунок 5.2 – текущая история коммитов ветки develop

На рисунке 6.1 представлена диаграмма пакетов

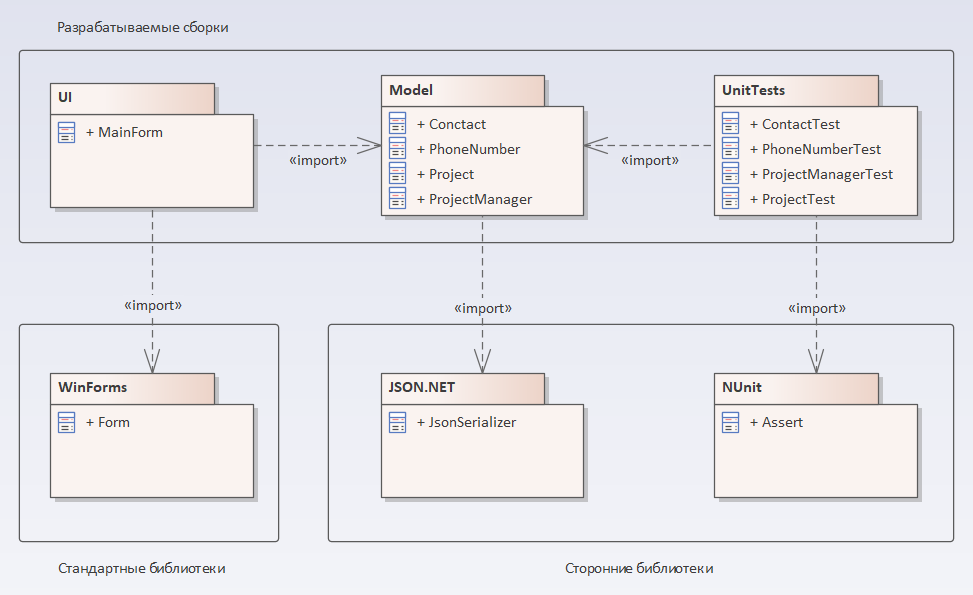
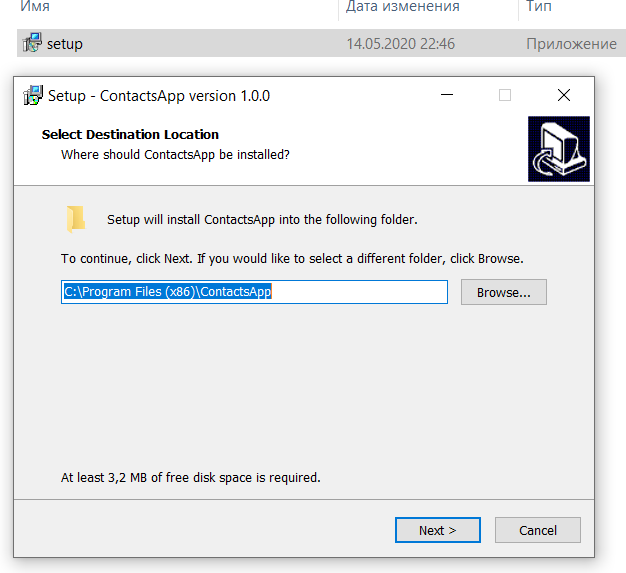


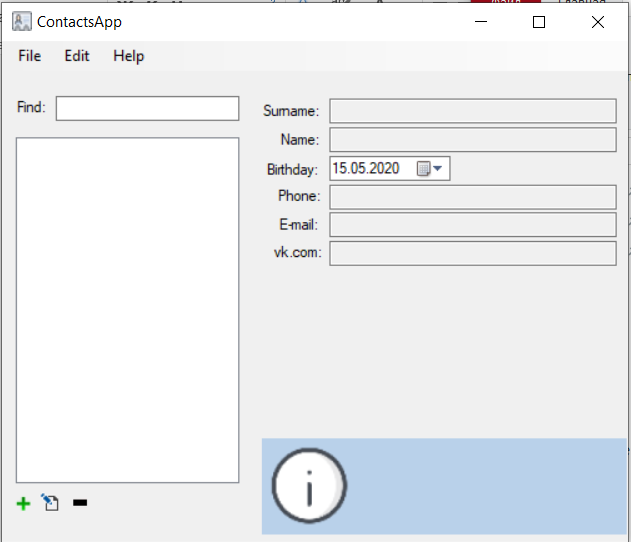
Рисунок 6.1 – диграмма пакетов

**Приемочное тестирование**

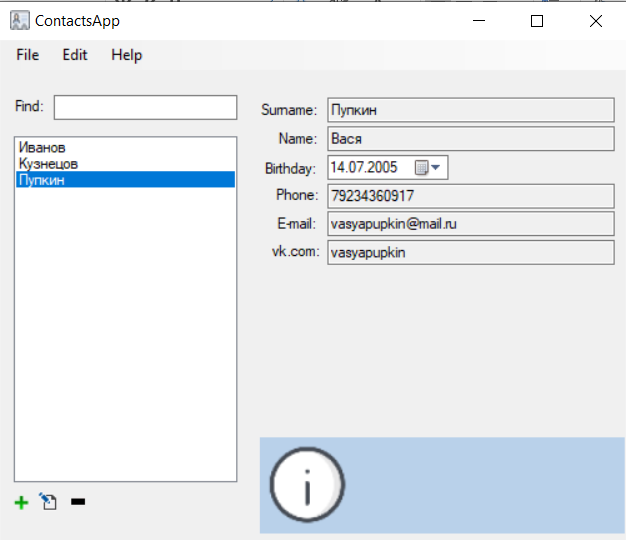
1. Установите приложение на компьютер с помощью собранного установочного пакета.



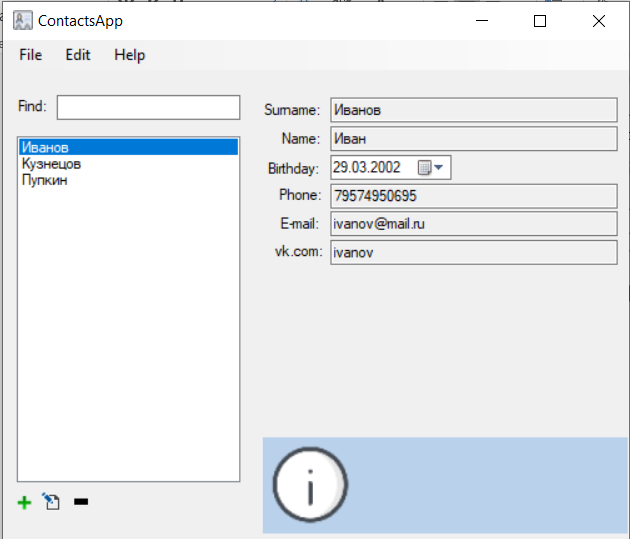
1. Запустите приложение. Окно программы должно быть пустым – в приложении не должно быть контактов.



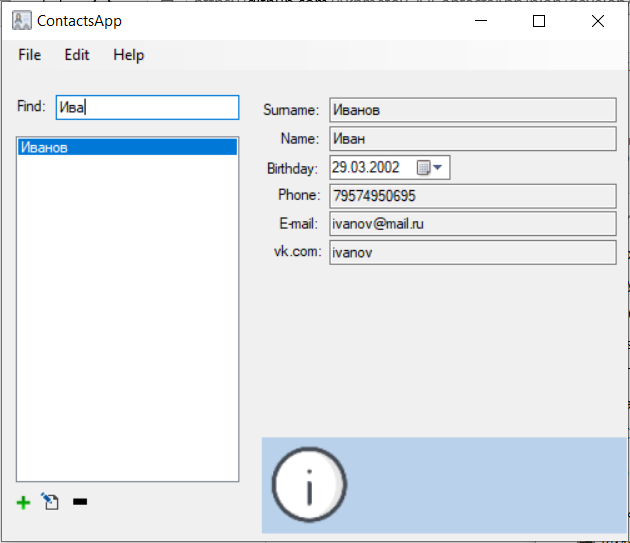
1. Создайте три контакта в приложении с разными фамилиями



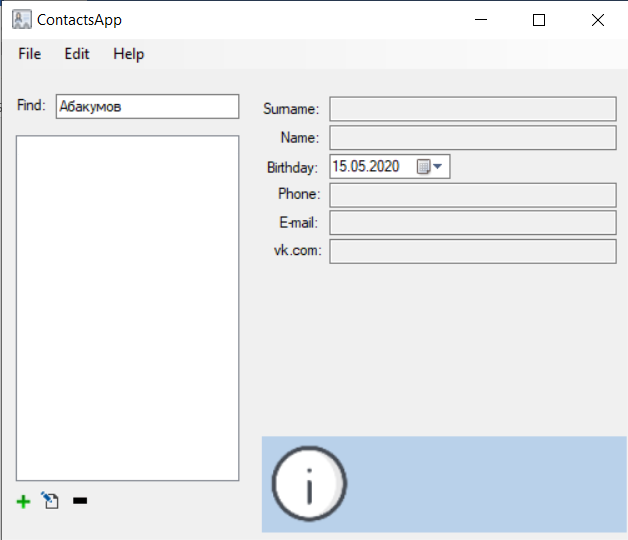
1. Переключитесь между контактами, показав, что смена текущего контакта в правой панели происходит корректно.



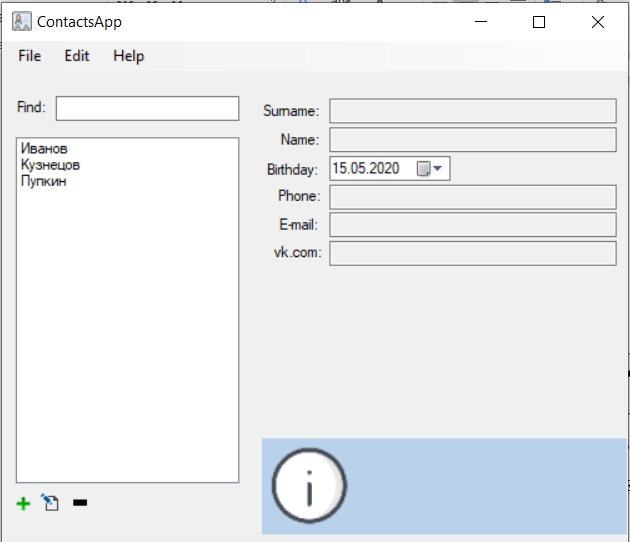
1. Введите в поиск подстроку для поиска контактов – в списке контактов должны остаться только контакты, содержащие подстроку.



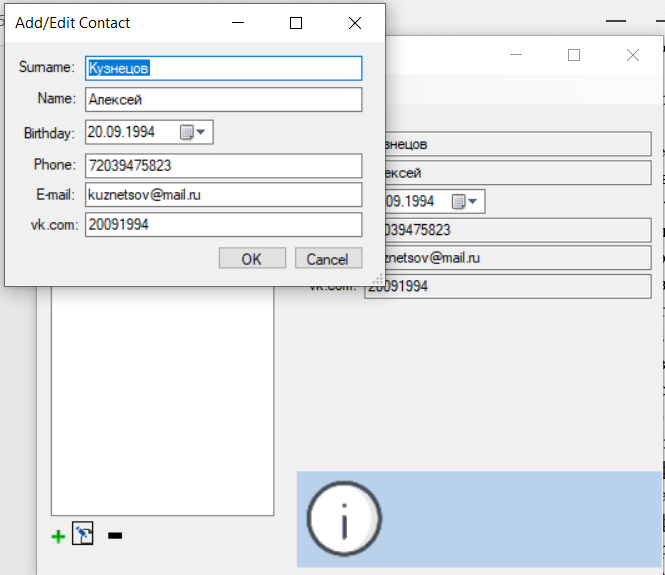
1. Введите в поиск подстроку, которой нет в фамилиях и именах контактов – список контактов должен быть пустым.



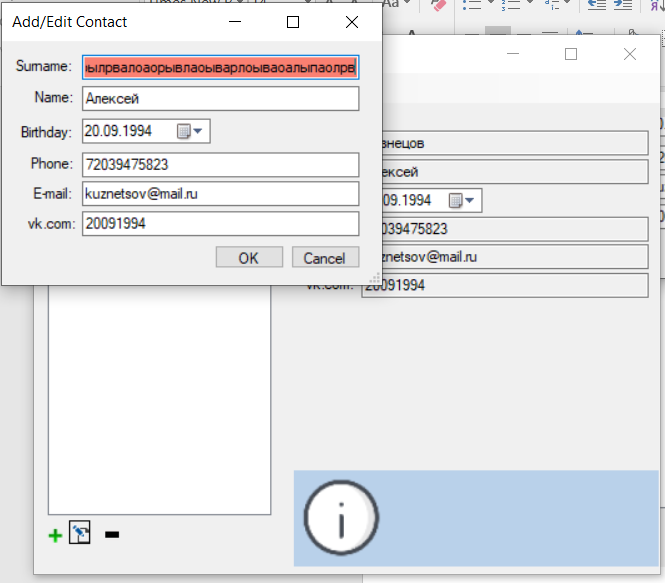
Сотрите подстроку поиска – список контактов должен восстановиться.



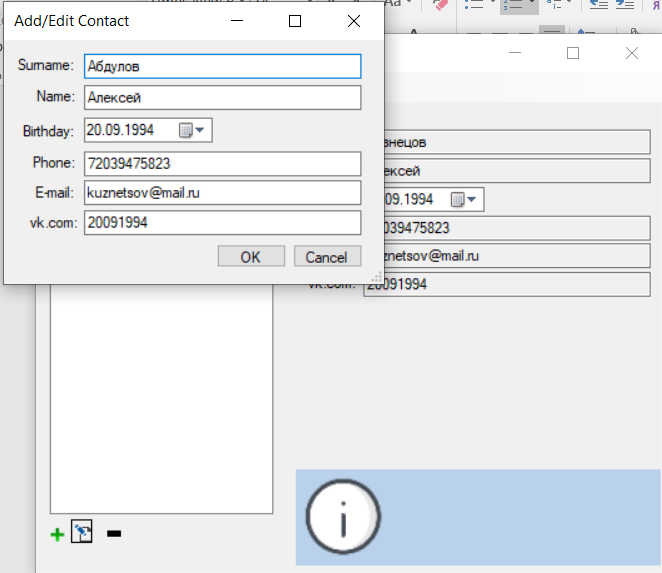
1. Выберите любой контакт и нажмите кнопку редактирования. Должно открыться окно редактирования контакта.



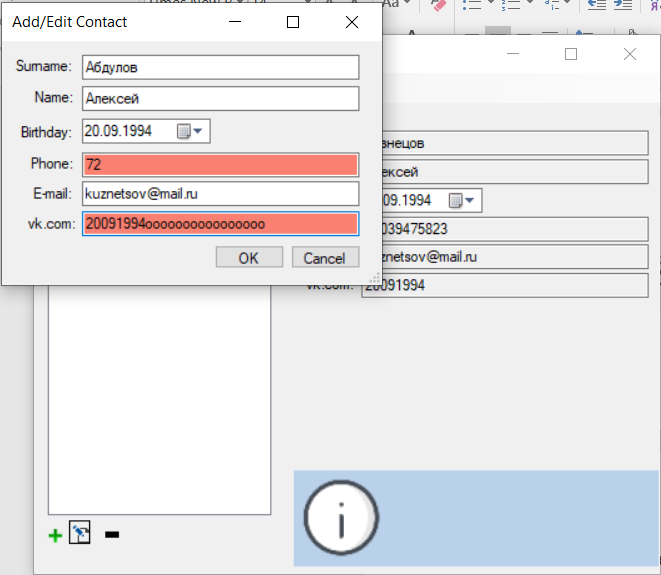
1. Введите фамилию более 50 символов. Элемент управления с фамилией должен указать на некорректное значение.



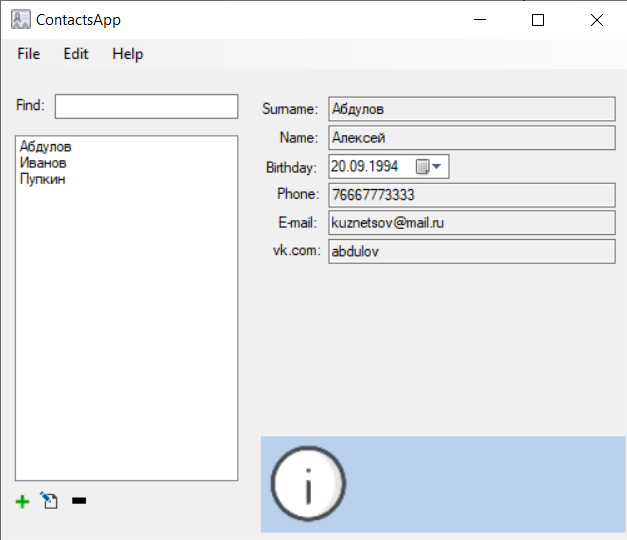
1. Введите фамилию менее 50 символов. Элемент управления с фамилией должен стать корректным.



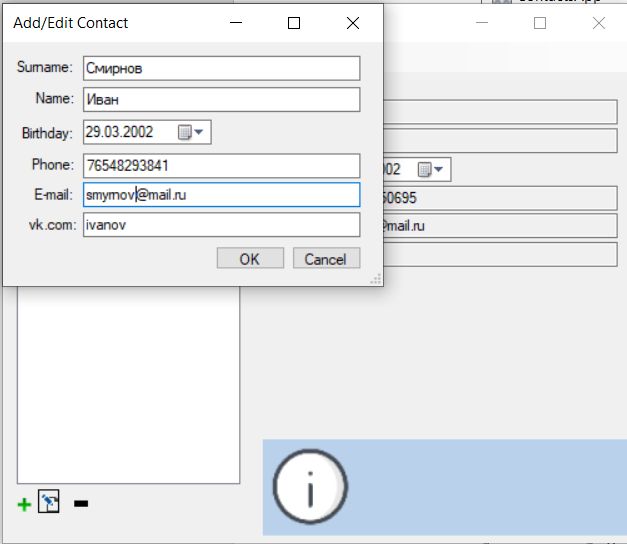
1. Покажите, что защита от некорректных значений также работает и для других полей.



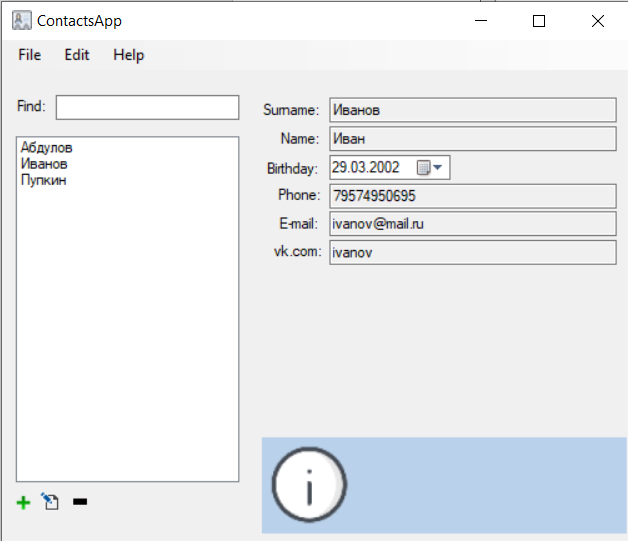
1. Поменяйте фамилию контакта, отличную от исходной. Поменяйте номер телефона. Нажмите «OK». Отредактированный контакт должен переместиться в списке контактов согласно алфавиту, отображаемые данные текущего контакта в правой панели также должны измениться.



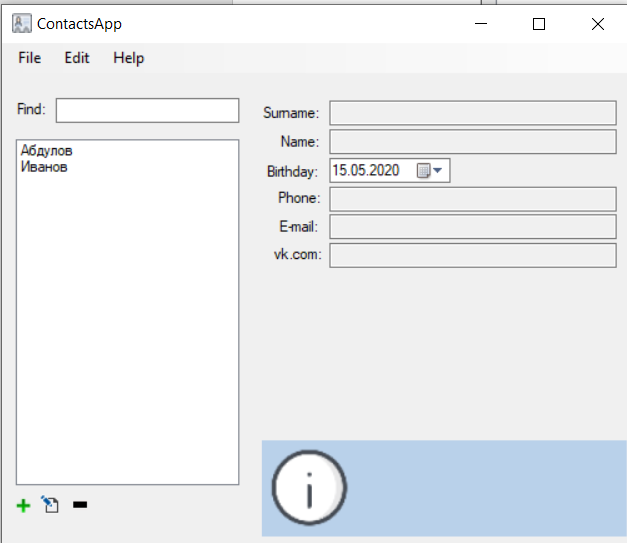
1. Выберите любой контакт и нажмите кнопку редактирования. Должно открыться окно редактирования. Измените фамилию контакта, номер телефона и e-mail.



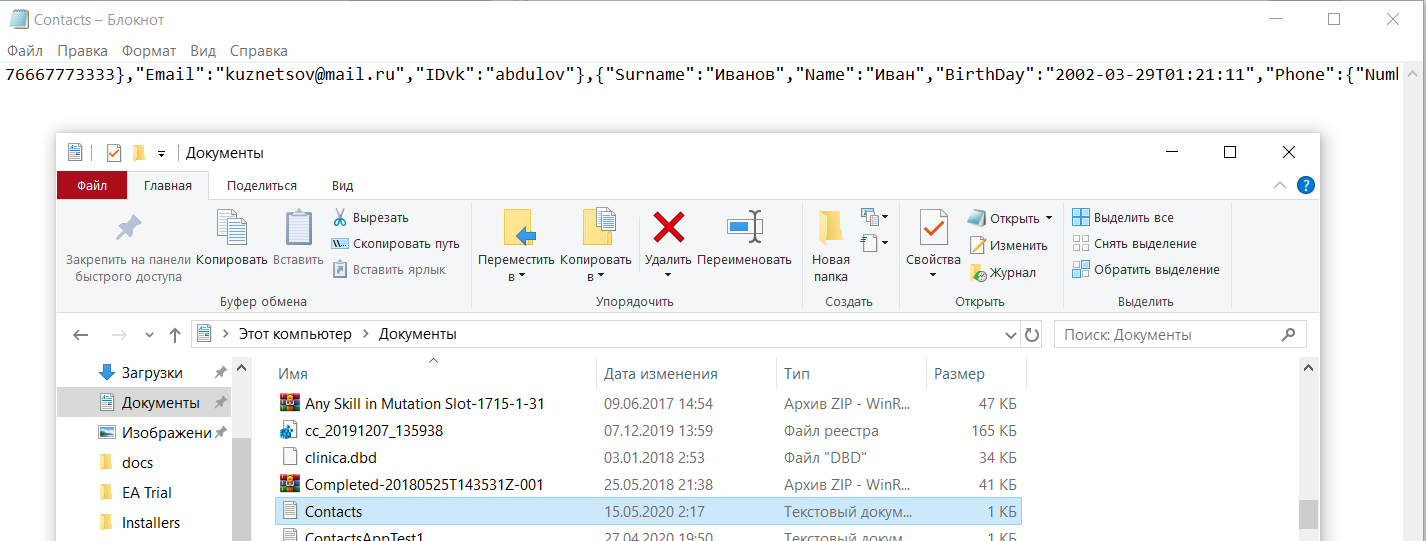
Нажмите «Cancel». Исходный контакт должен остаться без изменений.



1. Удалите третий контакт.



1. Закройте приложение. Должно произойти сохранение контактов в целевой файл.



1. Запустите приложение. В программе должны восстановиться контакты, созданные в предыдущую сессию.

