Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

”Алтайский государственный технический университет им. И. И. Ползунова”

Факультет информационных технологий

*наименование подразделения*

Кафедра прикладной математики

*наименование кафедры*

Отчет защищен с оценкой\_\_\_\_\_\_\_\_\_

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2018г.

Руководитель от вуза

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/**

*подпись Ф. И. О.*

**ОТЧЕТ**

О 1-ой учебной практике

Разработка программного обеспечения для получения навыков работы с графическим интерфейсом в среде разработки Microsoft Visual Studio

*общая формулировка задания*

в АлтГТУ

*наименование организации*

Студент гр. ПИ-72\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Приходько П.С.

*индекс группы подпись Ф. И. О.*

Руководитель от организации \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Егорова Е.В.

*подпись Ф. И. О.*

Руководитель от университета \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Егорова Е.В.

*подпись Ф. И. О.*

2018

ФГБОУ ВО «Алтайский государственный технический университет

им. И. И. Ползунова»

Кафедра \_\_Прикладная математика

**Индивидуальное задание**

на 1-ую учебную практику

(вид, тип и содержательная характеристика практики по УП)

студенту 1 курса \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ группы ПИ72

(Ф.И.О.)

Профильная организация\_\_АлтГТУ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование)

Сроки практики 25.06.2018 г. по 08.07.2018 г.

( по приказу АлтГТУ)

Тема Разработка программного обеспечения для получения навыков работы с графическим интерфейсом в среде разработки Microsoft Visual Studio

**Рабочий график (план) проведения практики:**

|  |  |  |  |
| --- | --- | --- | --- |
| **№**  **п/п** | **Содержание раздела (этапа)**  **практики** | **Сроки выполнения** | **Планируемые результаты практики** |
| 1 | Изучение технологии разработки графических интерфейсов в среде разработки Microsoft Visual Studio | 1 неделя | Формирование компетенции ОПК-1: владение основными концепциями, принципами, теориями и фактами, связанными с информатикой |
| 2 | Знакомство с MSDN. Самостоятельная работа с документацией | 1 неделя | Формирование компетенции ОК-7: способность к самоорганизации и самообразованию |
| 3 | Разработка приложения с графическим интерфейсом по индивидуальному заданию | 2 неделя | Формирование компетенции ПК-1: готовность применять основные методы и инструменты разработки программного обеспечения |
| 4 | Оформление и сдача отчета по практике. Знакомство с UML. | 2 неделя | Формирование компетенции ПК-15: способность готовить презентации, оформлять научно-технические отчеты по результатам выполненной работы, публиковать результаты исследований в виде статей и докладов на научно-технических конференциях |

|  |  |
| --- | --- |
| Руководитель практики от университета \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Егорова Е.В.  (подпись) (Ф.И.О., должность) |  |
| Руководитель практики от  профильной организации \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Егорова Е.В.  (подпись) (Ф.И.О., должность) |  |

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (Ф.И.О.)

Содержание

[**Введение** 4](#_Toc517946549)

[**1 Техническое задание** 5](#_Toc517946550)

[**1.1** **Задание** 5](#_Toc517946551)

[**1.2 Терминология** 5](#_Toc517946552)

[**1.3** **Требования к функциональности** 6](#_Toc517946553)

[**2 Проект программного продукта** 7](#_Toc517946554)

[**2.1 Структура данных** 7](#_Toc517946555)

[**2.2 Структура файлов** 8](#_Toc517946556)

[**3 Описание программного продукта** 9](#_Toc517946557)

[**3.1 Общий подход к программной реализации** 9](#_Toc517946558)

[**3.2 Модульная структура программного обеспечения** 9](#_Toc517946559)

[**3.3 Описание реализации** 12](#_Toc517946560)

[**Заключение** 15](#_Toc517946561)

[**Список использованных источников** 16](#_Toc517946562)

[**Приложение А Снимки экранных форм пользовательского интерфейса** 17](#_Toc517946563)

[**Приложение Б Исходный код программы** 21](#_Toc517946564)

[**Приложение В Руководство системного администратора** 78](#_Toc517946565)

[**Приложение Г Руководство пользователя** 79](#_Toc517946566)

# **Введение**

В настоящее время, обработка больших баз данных обретает все большую значимость.

Для работы с большими объемами данных разрабатываются приложения, упрощающие данную задачу. Эти программы делают обработку данных удобнее, быстрее и эффективнее. Такие приложения должны иметь удобный и простой интерфейс для понижения «порога вхождения» в данную систему работы.

Разработка приложений для работы с базами данных является необходимым и полезным навыком для программиста.

**Цель работы**: изучение принципов разработки графических приложений, получение навыков разработки приложений с графическим интерфейсом в среде Microsoft Visual Studio.

**Задача:** реализация приложения для работы со списком информации об абонентах, обладающего дружественным интерфейсом и позволяющим организовать эффективную обработку информации.

# **Техническое задание**

## **Задание**

Организовать работу со списком данных. Каждая строка списка содержит: Фамилия абонента; номер телефона; год установки; тип оплаты; адрес абонента(улица, дом, квартира). Например: Петров А.Н. 36-36-36 2001 НАЛ Ленина,54,102. Реализовать обработку файла со списком абонентов (добавление, удаление, редактирование записей, выполнение запросов, сохранение данных в файл) используя среду разработки Visual Studio и визуальные компоненты Windows Forms C++. Дополнительно: разработка игры.

## **1.2 Терминология**

Запись – строка, содержащая информацию об абоненте

Фамилия – строка, состоящая из символов

Год – строка, состоящая из цифр

Номер телефона – строка, состоящая из 6 цифр в виде (\_ \_)-\_ \_-\_ \_

Тип оплаты – строка принимающая 2 значения: НАЛ (наличными) и КАР (по карте)

Адрес – строка, состоящая из символов в виде: <Город>, <Улица>, <Номер дома>

Таблица – содержит список записей

Добавление записей – добавление записи в таблицу

Удаление – удаление записи из таблицы

Редактирование – редактирование записи и изменение ее в таблице

Сохранение – запись содержимого таблицы в файл

Запрос – поиск записей из таблицы, соответствующих данным в запросе

Поле – отдельное поле записи (тип оплаты, фамилия и т. д.)

## **Требования к функциональности**

Главная форма содержит таблицу с записями.

Возможности работы с таблицей:

* Открытие существующей таблицы (из файла)
* Сохранение изменений в существующей таблице (в файл)
* Создание новой таблицы

Возможности работы с записями:

* Добавление новых записей к существующим
* Редактирование полей уже существующих записей
* Удаление существующих записей

Возможности работы с запросами:

* Поиск записей соответствующих запросу

# **2 Проект программного продукта**

## **2.1 Структура данных**

В приложении были рализованы:

Формы:

* MainFrame – форма, в которой отображается таблица – dataGridView1. Из этой формы обеспечивается доступ к формам для добавления, редактирования, удаления записей, для запроса, для работы с таблицей, для формы с игрой, смены учетной записи, выхода и формы с общей информацией о программе. Также в этой форме отображается местонахождение текущего файла и уровень привилегий учетной записи.
* Request – форма, в которой происходит работа с запросами. В ней же контролируются введенные данные, при неудачном вводе, выдаются сообщения об ошибке
* AddingForm – форма, из которой происходит добавление записей в таблицу. В ней же производится проверка введенных данных на корректность. При неправильном вводе, выдаются сообщения об ошибке.
* EditingForm – форма, в которой происходит редактирование полей отдельных записей. В нее загружаются данные, которые в текущий момент соответствуют выделенной записи, а после редактирования, запись обновляется в таблице. В ней же происходит проверка введенных данных на корректность
* GameMetalSmith – форма, реализующая игру “Кузнец”.
* About – форма, представляющая общую информацию о программе и разработчиках
* Access – форма, реализующая доступ к программе, с помощью учетных записей. Имеет поля ввода логина и пароля, а так же доступ к форме для созданий новой учетной записи. В этой форме происходит проверка на наличие учетной записи в базе хранения учетных записей и проверку на правильность ввода и выдачу ошибок при их несоблюдении.
* CreateNewAccount – форма, реализующая создание новых учетных записей, и занесение их в базу хранения учетных записей. Проверяет на корректность введенные данные и выдает сообщения об ошибках в случае необходимости

## **2.2 Структура файлов**

Для работы программы необходимы: файл с именем AccessAccounts.txt, в который при работе будут заноситься сведения о существующих учетных записях, папка Sounds с вложенными в нее файлами со звуковым сопровождением игры и исполняемый файл программы. При работе возможно сохранить таблицы в формате (\*.txt).

# **3 Описание программного продукта**

**3.1 Общий подход к программной реализации**

Данное ПО реализовано на языке C++ в среде разработке программного обеспечения Microsoft Visual Studio 2017. Язык С++ - компилируемый статически типизированный язык программирования общего назначения.

Приложение соответствует всем современным парадигмам объектно-ориентированного программирования.

Для проверки введенных данных на корректность использовались элементы maskedTextBox, обработчики событий изменения текстового поля, обработчики событий нажатия кнопки. Для шифровки и дешифровки данных учетной записи(логин, пароль, тип пользователя) использовалось XOR шифрование и специальный ключ.

## **3.2 Модульная структура программного обеспечения**

В приложении реализованы следующие модули и взаимодействие между ними отображены на рисунке 3.1 и на диаграмме UML (рисунок 3.2)



Рисунок 3.1 – Модульная структура

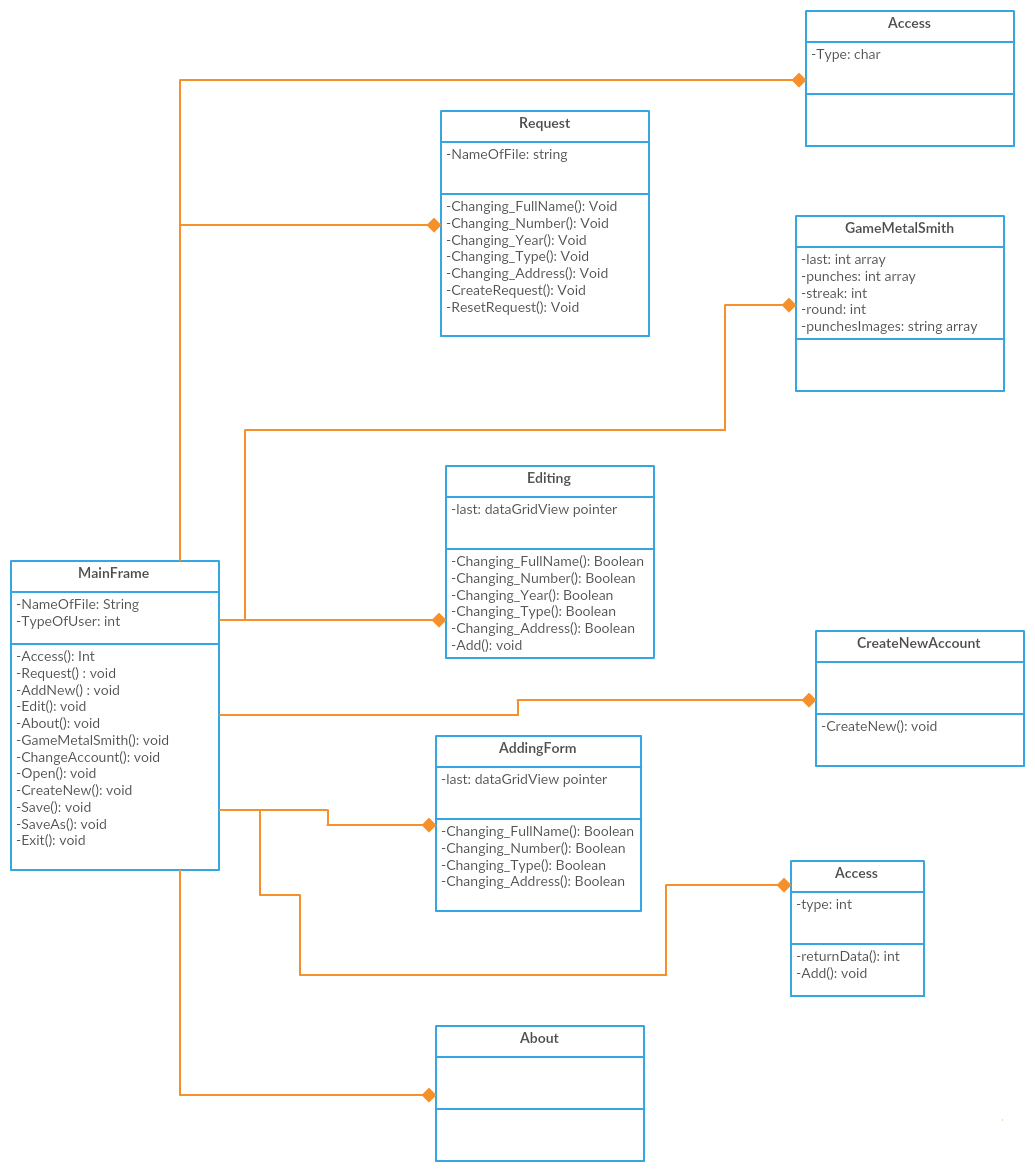


Рисунок 3.2 - Диаграмма классов

## **3.3 Описание реализации**

Главное окно(MainFrame) состоит из таблицы, для просмотра списка записей (dataGridView1) и ленты меню(toolStripBox). При загрузке приложения в конструкторе mainFrame производится скрытие (метод Hide()) этой формы и отображения (метод Showdialog) формы для входа в учетную запись(Access).

Форма Access состоит из 2 полей для ввода логина и пароля (элементы textbox), 2 кнопок – “Войти в учетную запись” и “Создать новую учетную запись” (элементы button) и надписи для вывода ошибок ( элемент label). Для контроля корректности введенных данных (на наличие ввода: “Введите пароль”, “Введите логин”) используется обработчик события “Изменение текста” (для textbox – textChanged). Если в одном из элементов textbox происходит изменение текста, то обработчик проверяет введенную строку на корректность и при необходимости блокирует доступ к кнопке “ Войти в учетную запись” и выдает сообщение об ошибке ввода. При нажатии на кнопку “Войти в учетную запись” введенные данные сверяются со списком имеющихся аккаунтов, зашифрованных и записанных в файл. При наличии в списке аналогичной записи, происходит скрытие формы и отображение формы MainFrame.

При нажатии на кнопку “Создать новую учетную запись” создается форма CreateNewAccount. Она содержит 2 поля для ввода логина и пароля, блок переключателей (элемент radioButton) и кнопки “Создать учетную запись”. При нажатии на кнопку “Создать учетную запись” производится проверка введенных данных на корректность и на наличие записей с аналогичным логином и, при необходимости, выводятся сообщения об ошибках. При соответствии введенных данным всем критериям, производится XOR шифрование логина и пароля и запись их в файл AccessAccounts.txt с помощью объекта класса StreamWriter пространства имен System::IO.

При отображении формы MainFrame из формы Access передается тип учетной записи в виде числа. Если это число 2 (соответствует уровню привилегий “Гость”) происходит скрытие блоков меню: Редактирование, и полей в меню: “Создать”, “Сохранить”, “Сохранить как”. При других значениях, все поля остаются доступными (уровень привилегий “Администратор”). Из этой формы предоставляется доступ ко всем остальным функциям приложения.

При выборе пункта меню “Создать” открывается окно создания файла (saveFileDialog), где вводится название файла и выбирается его месторасположение. После чего этот файл открывается, т.е. становится исполняемым файлом (его имя записывается в локальную переменную и используется при чтении данных и записи).

При выборе пункта меню “Открыть” открывается окно открытия файла (openFileDialog), где выбирается необходимый файл. Его имя записывается в локальную переменную и используется при чтении и записи. После чего происходит считывание строк из этого файла используя элемент StreamReader из пространства имен System::IO и добавление этих строк в dataGridView.

При выборе пункта меню “Сохранить” происходит считывание данных из dataGridView1 и запись этих данных в файл построчно (используя объект StreamWriter).

При выборе пункта меню “Сохранить как” происходит считывание данных из dataGridView1 и запись этих данных в файл, который выбирается в отдельном окне, построчно (используя объект StreamWriter). После чего этот файл становится исполняемым.

При выборе пункта меню “Запрос” происходит открытие формы Request. Форма Request содержит в себе таблицу dataGridView (в нее загружаются файлы из исполняемого файла, имя которого передается в конструктор как параметр), поля для ввода запроса (корректность введенных данных проверяется используя обработчик события textbox/maskedTextbox textextChange) блок элементов checkbox под каждым элементом textbox, для создания маски запроса и кнопки “Выполнить запрос” и “Сбросить запрос”. При нажатии кнопки “Выполнить запрос”, в соответствии с набором отмеченных элементов checkbox, при считывании очередной строки из файла, происходит сверка строк: считанной и занесенной в textbox (при помощи метода String::CompareTo()). При совпадении всех заданных полей, запись добавляется в dataGridView. При нажатии на кнопку “Сбросить запрос” происходит очистка dataGridView (Rows->Clear()) и повторная загрузка данных из файла.

При выборе пункта меню “Добавить запись” открывается форма AddingForm, которая имеет поля для ввода данных: элементы textbox и maskedTextBox (корректность введенных данных проверяется используя обработчик события textbox/maskedTextbox textextChange) и кнопки “Добавить запись”, при нажатии которой происходит конвертация содержимого текстовых полей в цельную строку (String) и добавление записи в dataGridView. В эту форму передается указатель на dataGridView из формы mainframe, и уже в AddingForm, используя указатель, применяется метод dataGridView->Rows->Add().

При выборе пункта меню “Редактировать” происходит открытие формы EditingForm. Она состоит из тех же полей, что и AddingForm, но в эти поля, используя указатель на dataGridView, заносятся значения, соответствующие полям этой записи в таблице. Все проверки производятся по аналогии с AddingForm. При нажатии на кнопку “Редактировать” происходит замена содержимого ячеек выделенной записи на содержимое полей ввода.

При выборе пункта меню “Удалить” выводится предупреждение об удалении записи. При положительном ответе происходит удаление выделенной записи в таблице, посредством метода RemoveAt().

При выборе пункта меню “О программе” выводится форма About, в которой в элементе textbox записана основная информация об эксплуатации приложения.

При выборе пункта меню “Игра” открывается форма GameMetalSmith. Состоит из 8 кнопок с ударами (каждая соответствует своему значению), из 2 элементов trackbar, на верхнем, при генерации уровня, задается положение ползунка, которое необходимо повторить, с помощью нажатий на кнопки, каждая из которых задает смещение положения ползунка на нижнем элементе trackbar, элемента progressBar, на котором с помощью элемента timer откладывается время до окончания раунда, и кнопки “Следующий раунд” которая и начинает новый уровень. По истечении времени выводится сообщение о проигрыше.

# **Заключение**

Программный продукт имеет простой и понятный интерфейс, не нуждающийся в изучении литературы или документации, и позволяет эффективно обрабатывать большие списки абонентов.

В процессе выполнения работы были изучены методы работы с объектами CLI, методы работы с файлами в языке C++, документация MSDN

Преимущества приложения:

* Интуитивно понятный интерфейс, не требующий изучения документации
* Высокая эффективность работы с записями
* Обработка уже созданных файлов и создание новых файлов
* Различные интерфейсы для режимов Администратора и пользователя
* Шифрование логина и пароля

Недостатки

* Работа только с одной таблицей
* Для работы необходимы специально отформатированные файлы
* Запрос работает только через логическое «И»

Возможные дальнейшие усовершенствования:

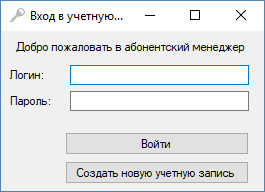
* Расширение набора обрабатываемых типов файлов
* Работа с базами данных SQL через Интернет
* Сохранение результатов запросов в файлы
* Добавление настроек интерфейса
* Расширение набора обрабатываемых видов таблиц
* Добавление возможности редактировать и удалять записи, полученные в результате запроса

# **Список использованных источников**

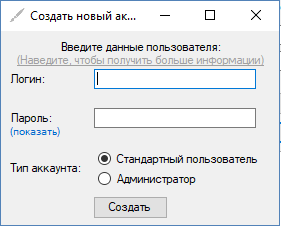
1. Егорова Е.В. Программирование на языке СИ. Учебное пособие/Алт. Госуд. Технич. Ун-т им. И.И.Ползунова.-Барнаул:2013.-184с.
2. Пахомов Б.И. C/C++ и MS Visual C++ 2010 для начинающих. – СПб.: БХВ-Петербург, 2011. – 736с.
3. “MSDN” – информационный сервис для разработчиков [Электронный ресурс] – Режим доступа http://msdn.microsoft.com/ru-ru/ms348103/library, свободный.

# **Приложение А Снимки экранных форм пользовательского интерфейса**

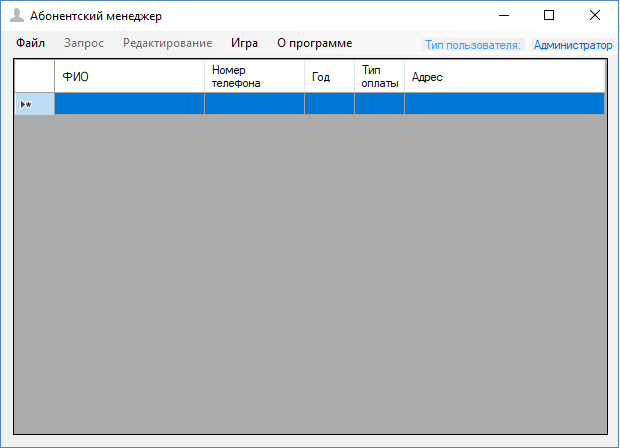
1. Вход в учетную запись



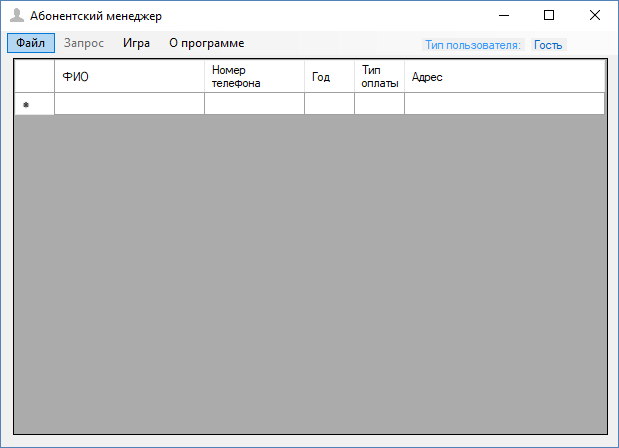
1. Создание нового аккаунта



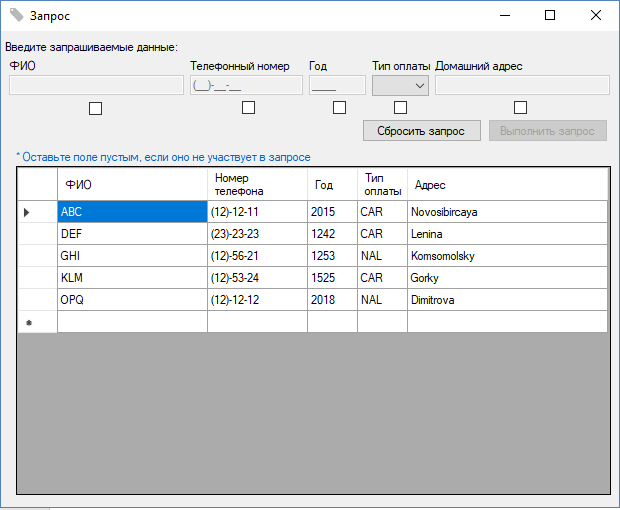
1. Основная область (просмотр списка) – Режим Администратора



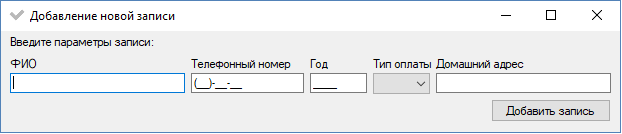
1. Основная область (просмотр списка) – Режим Гостя



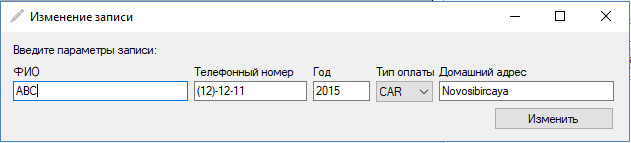
1. Окно запроса



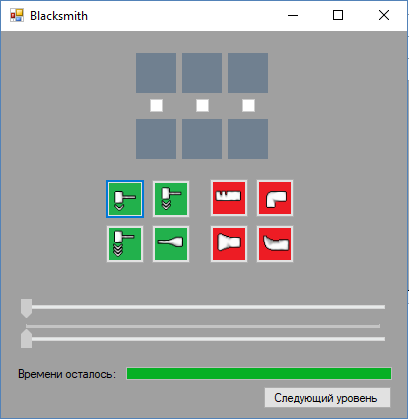
1. Добавление записи



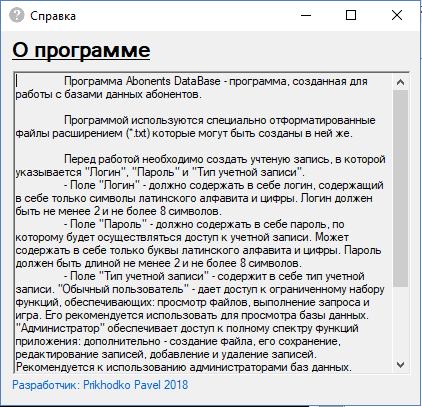
1. Редактирование выделенной записи



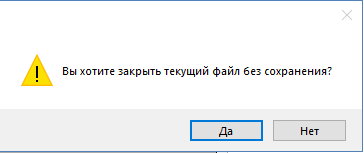
1. Игра



1. О программе



1. Предупреждение при закрытии



# **Приложение Б Исходный код программы**

MainFrame.h

#pragma once

#include "Access.h"

#include "Request.h"

#include "AddingForm.h"

#include "EditingForm.h"

#include "About.h"

#include "GameMetalSmith.h"

namespace Base {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class MainFrame : public System::Windows::Forms::Form

{

public:

MainFrame()

{

InitializeComponent();

Access ^access\_Window = gcnew Access;

Hide();

this->SaveToolStripMenuItem->Enabled = false;

this->SaveAsToolStripMenuItem->Enabled = false;

this->RequestToolStripMenuItem->Enabled = false;

this->EditionToolStripMenuItem->Enabled = false;

User->Text = "";

if(TypeOfUser = access\_Window->ShowDialog() == System::Windows::Forms::DialogResult::Cancel)

MainFrame::Close();

if (access\_Window->DialogResult == System::Windows::Forms::DialogResult::OK) {

if (access\_Window->ReturnData() == 2) {

this->CreateToolStripMenuItem->Visible = false;

this->SaveToolStripMenuItem->Visible = false;

this->SaveAsToolStripMenuItem->Visible = false;

this->EditionToolStripMenuItem->Visible = false;

User->Text = "Гость";

Show();

}

else

{

this->CreateToolStripMenuItem->Visible = true;

this->SaveToolStripMenuItem->Visible = true;

this->SaveAsToolStripMenuItem->Visible = true;

this->EditionToolStripMenuItem->Visible = true;

User->Text = "Администратор";

Show();

}

}

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MainFrame()

{

if (components)

{

delete components;

}

}

public: String ^NameOfFile;

private: int TypeOfUser;

public: System::Windows::Forms::DataGridView^ dataGridView1;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ fio;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ PhoneNumber;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Year;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ TypeOfPaying;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Adress;

private: System::Windows::Forms::MenuStrip^ menuStrip1;

private: System::Windows::Forms::ToolStripMenuItem^ FileToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ CreateToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ OpenToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ SaveAsToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ RequestToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ MakeRequestЗапросToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ EditionToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ gameToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ AboutToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ ChangeAccToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ ExitToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ DeliteToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ PlayToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ AboutInToolStripMenuItem1;

private: System::Windows::Forms::ToolStripMenuItem^ AddNewToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ EditToolStripMenuItem;

private: System::Windows::Forms::Label^ TypeOfAccount;

private: System::Windows::Forms::Label^ User;

private: System::Windows::Forms::ToolStripMenuItem^ SaveToolStripMenuItem;

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(MainFrame::typeid));

this->dataGridView1 = (gcnew System::Windows::Forms::DataGridView());

this->fio = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->PhoneNumber = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Year = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->TypeOfPaying = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Adress = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->menuStrip1 = (gcnew System::Windows::Forms::MenuStrip());

this->FileToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->CreateToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->OpenToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->SaveToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->SaveAsToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->ChangeAccToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->ExitToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->RequestToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->MakeRequestЗапросToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->EditionToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->AddNewToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->EditToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->DeliteToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->gameToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->PlayToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->AboutToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->AboutInToolStripMenuItem1 = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->TypeOfAccount = (gcnew System::Windows::Forms::Label());

this->User = (gcnew System::Windows::Forms::Label());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->BeginInit();

this->menuStrip1->SuspendLayout();

this->SuspendLayout();

//

// dataGridView1

//

this->dataGridView1->AllowUserToOrderColumns = true;

this->dataGridView1->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->dataGridView1->Columns->AddRange(gcnew cli::array< System::Windows::Forms::DataGridViewColumn^ >(5) {

this->fio, this->PhoneNumber,

this->Year, this->TypeOfPaying, this->Adress

});

this->dataGridView1->ImeMode = System::Windows::Forms::ImeMode::NoControl;

this->dataGridView1->Location = System::Drawing::Point(12, 27);

this->dataGridView1->Name = L"dataGridView1";

this->dataGridView1->ReadOnly = true;

this->dataGridView1->SelectionMode = System::Windows::Forms::DataGridViewSelectionMode::FullRowSelect;

this->dataGridView1->Size = System::Drawing::Size(595, 377);

this->dataGridView1->TabIndex = 0;

//

// fio

//

this->fio->HeaderText = L"ФИО";

this->fio->Name = L"fio";

this->fio->ReadOnly = true;

this->fio->Width = 150;

//

// PhoneNumber

//

this->PhoneNumber->HeaderText = L"Номер телефона";

this->PhoneNumber->Name = L"PhoneNumber";

this->PhoneNumber->ReadOnly = true;

//

// Year

//

this->Year->HeaderText = L"Год";

this->Year->Name = L"Year";

this->Year->ReadOnly = true;

this->Year->Width = 50;

//

// TypeOfPaying

//

this->TypeOfPaying->HeaderText = L"Тип оплаты";

this->TypeOfPaying->Name = L"TypeOfPaying";

this->TypeOfPaying->ReadOnly = true;

this->TypeOfPaying->Width = 50;

//

// Adress

//

this->Adress->HeaderText = L"Адрес";

this->Adress->Name = L"Adress";

this->Adress->ReadOnly = true;

this->Adress->Width = 200;

//

// menuStrip1

//

this->menuStrip1->Items->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(5) {

this->FileToolStripMenuItem,

this->RequestToolStripMenuItem, this->EditionToolStripMenuItem, this->gameToolStripMenuItem, this->AboutToolStripMenuItem

});

this->menuStrip1->Location = System::Drawing::Point(0, 0);

this->menuStrip1->Name = L"menuStrip1";

this->menuStrip1->Size = System::Drawing::Size(617, 24);

this->menuStrip1->TabIndex = 1;

this->menuStrip1->Text = L"menuStrip1";

//

// FileToolStripMenuItem

//

this->FileToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(6) {

this->CreateToolStripMenuItem,

this->OpenToolStripMenuItem, this->SaveToolStripMenuItem, this->SaveAsToolStripMenuItem, this->ChangeAccToolStripMenuItem, this->ExitToolStripMenuItem

});

this->FileToolStripMenuItem->Name = L"FileToolStripMenuItem";

this->FileToolStripMenuItem->Size = System::Drawing::Size(48, 20);

this->FileToolStripMenuItem->Text = L"Файл";

//

// CreateToolStripMenuItem

//

this->CreateToolStripMenuItem->Name = L"CreateToolStripMenuItem";

this->CreateToolStripMenuItem->Size = System::Drawing::Size(206, 22);

this->CreateToolStripMenuItem->Text = L"Создать";

this->CreateToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::создатьToolStripMenuItem\_Click);

//

// OpenToolStripMenuItem

//

this->OpenToolStripMenuItem->Name = L"OpenToolStripMenuItem";

this->OpenToolStripMenuItem->Size = System::Drawing::Size(206, 22);

this->OpenToolStripMenuItem->Text = L"Открыть";

this->OpenToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::открытьToolStripMenuItem\_Click);

//

// SaveToolStripMenuItem

//

this->SaveToolStripMenuItem->Name = L"SaveToolStripMenuItem";

this->SaveToolStripMenuItem->Size = System::Drawing::Size(206, 22);

this->SaveToolStripMenuItem->Text = L"Сохранить";

this->SaveToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::сохранитьToolStripMenuItem\_Click);

//

// SaveAsToolStripMenuItem

//

this->SaveAsToolStripMenuItem->Name = L"SaveAsToolStripMenuItem";

this->SaveAsToolStripMenuItem->Size = System::Drawing::Size(206, 22);

this->SaveAsToolStripMenuItem->Text = L"Сохранить как";

this->SaveAsToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::сохранитьВФайлToolStripMenuItem\_Click);

//

// ChangeAccToolStripMenuItem

//

this->ChangeAccToolStripMenuItem->Name = L"ChangeAccToolStripMenuItem";

this->ChangeAccToolStripMenuItem->Size = System::Drawing::Size(206, 22);

this->ChangeAccToolStripMenuItem->Text = L"Сменить учетню запись";

this->ChangeAccToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::сменитьУчетнюЗаписьToolStripMenuItem\_Click);

//

// ExitToolStripMenuItem

//

this->ExitToolStripMenuItem->Name = L"ExitToolStripMenuItem";

this->ExitToolStripMenuItem->Size = System::Drawing::Size(206, 22);

this->ExitToolStripMenuItem->Text = L"Выход";

this->ExitToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::выходToolStripMenuItem\_Click);

//

// RequestToolStripMenuItem

//

this->RequestToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(1) { this->MakeRequestЗапросToolStripMenuItem });

this->RequestToolStripMenuItem->Name = L"RequestToolStripMenuItem";

this->RequestToolStripMenuItem->Size = System::Drawing::Size(59, 20);

this->RequestToolStripMenuItem->Text = L"Запрос";

//

// MakeRequestЗапросToolStripMenuItem

//

this->MakeRequestЗапросToolStripMenuItem->Name = L"MakeRequestЗапросToolStripMenuItem";

this->MakeRequestЗапросToolStripMenuItem->Size = System::Drawing::Size(158, 22);

this->MakeRequestЗапросToolStripMenuItem->Text = L"Создать запрос";

this->MakeRequestЗапросToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::создатьЗапросToolStripMenuItem\_Click);

//

// EditionToolStripMenuItem

//

this->EditionToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(3) {

this->AddNewToolStripMenuItem,

this->EditToolStripMenuItem, this->DeliteToolStripMenuItem

});

this->EditionToolStripMenuItem->Name = L"EditionToolStripMenuItem";

this->EditionToolStripMenuItem->Size = System::Drawing::Size(108, 20);

this->EditionToolStripMenuItem->Text = L"Редактирование";

//

// AddNewToolStripMenuItem

//

this->AddNewToolStripMenuItem->Name = L"AddNewToolStripMenuItem";

this->AddNewToolStripMenuItem->Size = System::Drawing::Size(166, 22);

this->AddNewToolStripMenuItem->Text = L"Добавить запись";

this->AddNewToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::добавтьЗаписьToolStripMenuItem\_Click);

//

// EditToolStripMenuItem

//

this->EditToolStripMenuItem->Name = L"EditToolStripMenuItem";

this->EditToolStripMenuItem->Size = System::Drawing::Size(166, 22);

this->EditToolStripMenuItem->Text = L"Редактировать";

this->EditToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::редактироватьToolStripMenuItem\_Click);

//

// DeliteToolStripMenuItem

//

this->DeliteToolStripMenuItem->Name = L"DeliteToolStripMenuItem";

this->DeliteToolStripMenuItem->Size = System::Drawing::Size(166, 22);

this->DeliteToolStripMenuItem->Text = L"Удалить";

this->DeliteToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::удалитьToolStripMenuItem\_Click);

//

// gameToolStripMenuItem

//

this->gameToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(1) { this->PlayToolStripMenuItem });

this->gameToolStripMenuItem->Name = L"gameToolStripMenuItem";

this->gameToolStripMenuItem->Size = System::Drawing::Size(46, 20);

this->gameToolStripMenuItem->Text = L"Игра";

//

// PlayToolStripMenuItem

//

this->PlayToolStripMenuItem->Name = L"PlayToolStripMenuItem";

this->PlayToolStripMenuItem->Size = System::Drawing::Size(112, 22);

this->PlayToolStripMenuItem->Text = L"Играть";

this->PlayToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainFrame::PlayToolStripMenuItem\_Click);

//

// AboutToolStripMenuItem

//

this->AboutToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(1) { this->AboutInToolStripMenuItem1 });

this->AboutToolStripMenuItem->Name = L"AboutToolStripMenuItem";

this->AboutToolStripMenuItem->Size = System::Drawing::Size(94, 20);

this->AboutToolStripMenuItem->Text = L"О программе";

//

// AboutInToolStripMenuItem1

//

this->AboutInToolStripMenuItem1->Name = L"AboutInToolStripMenuItem1";

this->AboutInToolStripMenuItem1->Size = System::Drawing::Size(152, 22);

this->AboutInToolStripMenuItem1->Text = L"О программе";

this->AboutInToolStripMenuItem1->Click += gcnew System::EventHandler(this, &MainFrame::AboutInToolStripMenuItem1\_Click);

//

// TypeOfAccount

//

this->TypeOfAccount->AutoSize = true;

this->TypeOfAccount->ForeColor = System::Drawing::SystemColors::MenuHighlight;

this->TypeOfAccount->Location = System::Drawing::Point(421, 7);

this->TypeOfAccount->Name = L"TypeOfAccount";

this->TypeOfAccount->Size = System::Drawing::Size(103, 13);

this->TypeOfAccount->TabIndex = 2;

this->TypeOfAccount->Text = L"Тип пользователя:";

//

// User

//

this->User->AutoSize = true;

this->User->ForeColor = System::Drawing::SystemColors::HotTrack;

this->User->Location = System::Drawing::Point(530, 7);

this->User->Name = L"User";

this->User->Size = System::Drawing::Size(36, 13);

this->User->TabIndex = 2;

this->User->Text = L"Гость";

//

// MainFrame

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(617, 416);

this->Controls->Add(this->User);

this->Controls->Add(this->TypeOfAccount);

this->Controls->Add(this->dataGridView1);

this->Controls->Add(this->menuStrip1);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedDialog;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->MainMenuStrip = this->menuStrip1;

this->Name = L"MainFrame";

this->Text = L"Абонентский менеджер";

this->FormClosing += gcnew System::Windows::Forms::FormClosingEventHandler(this, &MainFrame::MainFrame\_FormClosing);

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->EndInit();

this->menuStrip1->ResumeLayout(false);

this->menuStrip1->PerformLayout();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void сменитьУчетнюЗаписьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Access ^changeAccount = gcnew Access;

Hide();

changeAccount->ShowDialog();

if (changeAccount->DialogResult == System::Windows::Forms::DialogResult::OK) {

dataGridView1->Rows->Clear();

if (changeAccount->ReturnData() == 2) {

this->CreateToolStripMenuItem->Visible = false;

this->SaveToolStripMenuItem->Visible = false;

this->SaveAsToolStripMenuItem->Visible = false;

this->EditionToolStripMenuItem->Visible = false;

User->Text = "Гость";

}

else

{

this->CreateToolStripMenuItem->Visible = true;

this->SaveToolStripMenuItem->Visible = true;

this->SaveAsToolStripMenuItem->Visible = true;

this->EditionToolStripMenuItem->Visible = true;

User->Text = "Администратор";

}

}

Show();

}

private: System::Void создатьЗапросToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Request ^newRequest = gcnew Request(NameOfFile);

newRequest->ShowDialog();

}

private: System::Void добавтьЗаписьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

AddingForm ^newLine = gcnew AddingForm(dataGridView1);

newLine->ShowDialog();

}

private: System::Void открытьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

OpenFileDialog ^openedFile = gcnew OpenFileDialog;

openedFile->Filter = "Text files(\*.txt)|\* txt|All files(\*.\*)|\*.\*";

if(dataGridView1->RowCount > 1){

if (MessageBox::Show("Вы верены, что хотите закрыть текущее расписание?", "Base",

MessageBoxButtons::YesNo, MessageBoxIcon::Warning) == System::Windows::Forms::DialogResult::Yes) {

if (openedFile->ShowDialog() == System::Windows::Forms::DialogResult::OK) {

dataGridView1->Rows->Clear();

NameOfFile = openedFile->FileName;

StreamReader ^tempFile = gcnew StreamReader(NameOfFile);

String ^row;

while (tempFile->Peek() >= 0) {

row = tempFile->ReadLine();

array <String^>^ cells = row->Split(' ', '\0');

if (cells[0]->CompareTo(""))

dataGridView1->Rows->Add(cells);

}

this->SaveToolStripMenuItem->Enabled = true;

this->SaveAsToolStripMenuItem->Enabled = true;

this->RequestToolStripMenuItem->Enabled = true;

this->EditionToolStripMenuItem->Enabled = true;

tempFile->Close();

this->Text = "Абонентский менеджер" + ": " + NameOfFile;

}

}

}

else {

if (openedFile->ShowDialog() == System::Windows::Forms::DialogResult::OK) {

dataGridView1->Rows->Clear();

NameOfFile = openedFile->FileName;

StreamReader ^tempFile = gcnew StreamReader(NameOfFile);

String ^row;

while (tempFile->Peek() >= 0) {

row = tempFile->ReadLine();

array <String^>^ cells = row->Split(' ','\0');

if(cells[0]->CompareTo(""))

dataGridView1->Rows->Add(cells);

}

tempFile->Close();

this->SaveToolStripMenuItem->Enabled = true;

this->SaveAsToolStripMenuItem->Enabled = true;

this->RequestToolStripMenuItem->Enabled = true;

this->EditionToolStripMenuItem->Enabled = true;

tempFile->Close();

this->Text = "MainFrame" + ": " + NameOfFile;

}

}

}

private: System::Void создатьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

SaveFileDialog ^creatingFile = gcnew SaveFileDialog;

creatingFile->Filter = "Text files(\*.txt) | \*txt";

if (dataGridView1->RowCount > 1) {

if (MessageBox::Show("Вы верены, что хотите закрыть текущее расписание?", "Base",

MessageBoxButtons::YesNo, MessageBoxIcon::Warning) == System::Windows::Forms::DialogResult::Yes) {

if (creatingFile->ShowDialog() == System::Windows::Forms::DialogResult::OK) {

dataGridView1->Rows->Clear();

NameOfFile = creatingFile->FileName;

FileStream ^newFile = File::Create(NameOfFile);

newFile->Close();

this->SaveToolStripMenuItem->Enabled = true;

this->SaveAsToolStripMenuItem->Enabled = true;

this->RequestToolStripMenuItem->Enabled = true;

this->EditionToolStripMenuItem->Enabled = true;

}

}

}

else {

if (creatingFile->ShowDialog() == System::Windows::Forms::DialogResult::OK) {

dataGridView1->Rows->Clear();

NameOfFile = creatingFile->FileName;

FileStream ^newFile = File::Create(NameOfFile);

newFile->Close();

this->SaveToolStripMenuItem->Enabled = true;

this->SaveAsToolStripMenuItem->Enabled = true;

this->RequestToolStripMenuItem->Enabled = true;

this->EditionToolStripMenuItem->Enabled = true;

}

}

}

private: System::Void сохранитьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

FileStream ^RewritedFile = File::Create(NameOfFile);

RewritedFile->Close();

StreamWriter ^Writer = gcnew StreamWriter(NameOfFile);

for (int i = 0; i < dataGridView1->RowCount - 1; i++) {

Writer->Write(dataGridView1->Rows[i]->Cells[0]->Value);

Writer->Write(" ");

Writer->Write(dataGridView1->Rows[i]->Cells[1]->Value);

Writer->Write(" ");

Writer->Write(dataGridView1->Rows[i]->Cells[2]->Value);

Writer->Write(" ");

Writer->Write(dataGridView1->Rows[i]->Cells[3]->Value);

Writer->Write(" ");

Writer->Write(dataGridView1->Rows[i]->Cells[4]->Value);

Writer->WriteLine();

}

Writer->Close();

}

private: System::Void сохранитьВФайлToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

SaveFileDialog ^creatingFile = gcnew SaveFileDialog;

creatingFile->Filter = "Text files(\*.txt) | \*txt";

if (creatingFile->ShowDialog() == System::Windows::Forms::DialogResult::OK) {

NameOfFile = creatingFile->FileName;

FileStream ^RewritedFile = File::Create(NameOfFile);

RewritedFile->Close();

StreamWriter ^Writer = gcnew StreamWriter(NameOfFile);

for (int i = 0; i < dataGridView1->RowCount - 1; i++) {

Writer->Write(dataGridView1->Rows[i]->Cells[0]->Value);

Writer->Write(" ");

Writer->Write(dataGridView1->Rows[i]->Cells[1]->Value);

Writer->Write(" ");

Writer->Write(dataGridView1->Rows[i]->Cells[2]->Value);

Writer->Write(" ");

Writer->Write(dataGridView1->Rows[i]->Cells[3]->Value);

Writer->Write(" ");

Writer->Write(dataGridView1->Rows[i]->Cells[4]->Value);

Writer->WriteLine();

}

Writer->Close();

}

this->Text = "MainFrame" + ": " + NameOfFile;

}

private: System::Void выходToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

MainFrame::Close();

}

private: System::Void редактироватьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

array <String^>^ tempRow = { (dataGridView1->Rows[dataGridView1->CurrentRow->Index]->Cells[0]->Value)->ToString(),

(dataGridView1->Rows[dataGridView1->CurrentRow->Index]->Cells[1]->Value)->ToString(),

(dataGridView1->Rows[dataGridView1->CurrentRow->Index]->Cells[2]->Value)->ToString(),

(dataGridView1->Rows[dataGridView1->CurrentRow->Index]->Cells[3]->Value)->ToString(),

(dataGridView1->Rows[dataGridView1->CurrentRow->Index]->Cells[4]->Value)->ToString() };

EditingForm ^addNew = gcnew EditingForm(dataGridView1);

addNew->ShowDialog();

}

private: System::Void удалитьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

if (MessageBox::Show("Вы уверены, что хотите удалить эту запись?", "",

MessageBoxButtons::YesNo, MessageBoxIcon::Warning) == System::Windows::Forms::DialogResult::Yes) {

dataGridView1->Rows->RemoveAt(dataGridView1->CurrentRow->Index);

dataGridView1->Refresh();

}

}

private: System::Void AboutInToolStripMenuItem1\_Click(System::Object^ sender, System::EventArgs^ e) {

About ^AboutBase = gcnew About();

AboutBase->ShowDialog();

}

private: System::Void PlayToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

GameMetalSmith ^game = gcnew GameMetalSmith();

game->Show();

}

private: System::Void MainFrame\_FormClosing(System::Object^ sender, System::Windows::Forms::FormClosingEventArgs^ e) {

if ((TypeOfUser == 0) && (NameOfFile != \_\_nullptr)) {

if (MessageBox::Show("Вы хотите закрыть текущий файл без сохранения?", "",

MessageBoxButtons::YesNo, MessageBoxIcon::Warning) == System::Windows::Forms::DialogResult::No) {

e->Cancel = true;

}

}

}

};

}

MainFrame.h

#pragma once

#include "CreateNewAccount.h"

namespace Base {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

/// <summary>

/// Сводка для Access

/// </summary>

public ref class Access : public System::Windows::Forms::Form

{

public:

Access()

{

InitializeComponent();

}

public:Int32 ReturnData() {

if (type == 'A')

return 1;

if (type == 'U')

return 2;

return 0;

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~Access()

{

if (components)

{

delete components;

}

}

private: Char type;

private: System::Windows::Forms::TextBox^ textBox\_Access\_Login;

private: System::Windows::Forms::TextBox^ textBox\_Access\_Password;

private: System::Windows::Forms::Button^ button\_Access\_Enter;

private: System::Windows::Forms::Label^ label1\_Access\_Login;

private: System::Windows::Forms::Label^ label\_Access\_Password;

private: System::Windows::Forms::Label^ label\_Access\_Description;

private: System::Windows::Forms::Label^ Error;

private: System::Windows::Forms::Button^ button1;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(Access::typeid));

this->textBox\_Access\_Login = (gcnew System::Windows::Forms::TextBox());

this->textBox\_Access\_Password = (gcnew System::Windows::Forms::TextBox());

this->button\_Access\_Enter = (gcnew System::Windows::Forms::Button());

this->label1\_Access\_Login = (gcnew System::Windows::Forms::Label());

this->label\_Access\_Password = (gcnew System::Windows::Forms::Label());

this->label\_Access\_Description = (gcnew System::Windows::Forms::Label());

this->Error = (gcnew System::Windows::Forms::Label());

this->button1 = (gcnew System::Windows::Forms::Button());

this->SuspendLayout();

//

// textBox\_Access\_Login

//

this->textBox\_Access\_Login->Location = System::Drawing::Point(69, 34);

this->textBox\_Access\_Login->Name = L"textBox\_Access\_Login";

this->textBox\_Access\_Login->Size = System::Drawing::Size(179, 20);

this->textBox\_Access\_Login->TabIndex = 0;

this->textBox\_Access\_Login->TextChanged += gcnew System::EventHandler(this, &Access::textBox\_Access\_Login\_TextChanged);

//

// textBox\_Access\_Password

//

this->textBox\_Access\_Password->Location = System::Drawing::Point(69, 60);

this->textBox\_Access\_Password->Name = L"textBox\_Access\_Password";

this->textBox\_Access\_Password->Size = System::Drawing::Size(179, 20);

this->textBox\_Access\_Password->TabIndex = 2;

this->textBox\_Access\_Password->UseSystemPasswordChar = true;

this->textBox\_Access\_Password->TextChanged += gcnew System::EventHandler(this, &Access::textBox\_Access\_Password\_TextChanged);

//

// button\_Access\_Enter

//

this->button\_Access\_Enter->Location = System::Drawing::Point(64, 101);

this->button\_Access\_Enter->Name = L"button\_Access\_Enter";

this->button\_Access\_Enter->Size = System::Drawing::Size(184, 23);

this->button\_Access\_Enter->TabIndex = 4;

this->button\_Access\_Enter->Text = L"Войти";

this->button\_Access\_Enter->UseVisualStyleBackColor = true;

this->button\_Access\_Enter->Click += gcnew System::EventHandler(this, &Access::button\_Access\_Enter\_Click);

//

// label1\_Access\_Login

//

this->label1\_Access\_Login->AutoSize = true;

this->label1\_Access\_Login->Location = System::Drawing::Point(6, 37);

this->label1\_Access\_Login->Name = L"label1\_Access\_Login";

this->label1\_Access\_Login->Size = System::Drawing::Size(41, 13);

this->label1\_Access\_Login->TabIndex = 2;

this->label1\_Access\_Login->Text = L"Логин:";

//

// label\_Access\_Password

//

this->label\_Access\_Password->AutoSize = true;

this->label\_Access\_Password->Location = System::Drawing::Point(6, 63);

this->label\_Access\_Password->Name = L"label\_Access\_Password";

this->label\_Access\_Password->Size = System::Drawing::Size(48, 13);

this->label\_Access\_Password->TabIndex = 2;

this->label\_Access\_Password->Text = L"Пароль:";

//

// label\_Access\_Description

//

this->label\_Access\_Description->AutoSize = true;

this->label\_Access\_Description->Location = System::Drawing::Point(12, 9);

this->label\_Access\_Description->Name = L"label\_Access\_Description";

this->label\_Access\_Description->Size = System::Drawing::Size(236, 13);

this->label\_Access\_Description->TabIndex = 3;

this->label\_Access\_Description->Text = L"Добро пожаловать в абонентский менеджер";

//

// Error

//

this->Error->AutoSize = true;

this->Error->ForeColor = System::Drawing::Color::Red;

this->Error->Location = System::Drawing::Point(66, 85);

this->Error->Name = L"Error";

this->Error->Size = System::Drawing::Size(0, 13);

this->Error->TabIndex = 4;

//

// button1

//

this->button1->Location = System::Drawing::Point(64, 130);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(184, 23);

this->button1->TabIndex = 6;

this->button1->Text = L"Создать новую учетную запись";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &Access::button1\_Click);

//

// Access

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(263, 160);

this->Controls->Add(this->button1);

this->Controls->Add(this->Error);

this->Controls->Add(this->label\_Access\_Description);

this->Controls->Add(this->label\_Access\_Password);

this->Controls->Add(this->label1\_Access\_Login);

this->Controls->Add(this->button\_Access\_Enter);

this->Controls->Add(this->textBox\_Access\_Password);

this->Controls->Add(this->textBox\_Access\_Login);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedDialog;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"Access";

this->Text = L"Вход в учетную запись";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void button\_Access\_Enter\_Click(System::Object^ sender, System::EventArgs^ e) {

array <Char>^ key = { 1,2,3,4,5,6,7,8,9 };

array <Char>^ login = textBox\_Access\_Login->Text->ToCharArray();

array <Char>^ password = textBox\_Access\_Password->Text->ToCharArray();

if (!textBox\_Access\_Login->Text->CompareTo("") || !textBox\_Access\_Password->Text->CompareTo("")) {

if (!textBox\_Access\_Login->Text->CompareTo(""))Error->Text = "Введите логин!";

else Error->Text = "Введите пароль!";

}

else {

for (int i = 0; i < login->Length; i++)

login[i] ^= key[i];

for (int i = 0; i < password->Length; i++)

password[i] ^= key[8 - i];

StreamReader ^accessFile = gcnew StreamReader("AccessAccounts.txt");

String ^row;

int isPasswordAccepted = 1, isLoginAccepted = 1, isHave = 0;

while (accessFile->Peek() >= 0) {

isPasswordAccepted = 1;

isLoginAccepted = 1;

row = accessFile->ReadLine();

array <String^>^ cells = row->Split(' ', '\0');

array <Char>^ loginNew = cells[0]->ToCharArray();

array <Char>^ passwordNew = cells[1]->ToCharArray();

if (login->Length == loginNew->Length) {

for (int i = 0; i < login->Length; i++)

if (login[i] != loginNew[i]) {

isLoginAccepted = 0;

break;

}

}

else continue;

if ((password->Length == passwordNew->Length) && isLoginAccepted) {

for (int i = 0; i < password->Length; i++)

if (password[i] != passwordNew[i]) {

isPasswordAccepted = 0;

break;

}

}

if (isLoginAccepted && isPasswordAccepted) {

type = (cells[2]->ToCharArray())[0];

isHave = 1;

break;

}

}

accessFile->Close();

if (isHave) {

this->DialogResult = System::Windows::Forms::DialogResult::OK;

}

else {

Error->Text = "Неправильный логин/пароль!";

}

}

}

private: System::Void textBox\_Access\_Login\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Error->Text = "";}

private: System::Void textBox\_Access\_Password\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Error->Text = "";}

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) {

CreateNewAccount ^CreateNew = gcnew CreateNewAccount();

CreateNew->Enabled = true;

CreateNew->ShowDialog();}

};

}

CreateNewAccount.h

#pragma once

namespace Base {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

/// <summary>

/// Сводка для CreateNewAccount

/// </summary>

public ref class CreateNewAccount : public System::Windows::Forms::Form

{

public:

CreateNewAccount()

{

InitializeComponent();

ToolTip ^About = gcnew ToolTip;

About->SetToolTip(this->label5, "Логин и пароль могут содержать только цифры и буквы \nлатинского алфавита и быть длиной от 2 до 8 символов.\n ");

About->InitialDelay = 50;

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~CreateNewAccount()

{

if (components)

{

delete components;

}

}

protected:

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::TextBox^ textBox1\_Login;

private: System::Windows::Forms::TextBox^ textBox2\_Password;

private: System::Windows::Forms::Label^ ErrorLogin;

private: System::Windows::Forms::Label^ ErrorPassword;

private: System::Windows::Forms::RadioButton^ radioButton1;

private: System::Windows::Forms::RadioButton^ radioButton2;

private: System::Windows::Forms::Panel^ panel1;

private: System::Windows::Forms::Button^ CreateAccount;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Label^ label6;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(CreateNewAccount::typeid));

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->textBox1\_Login = (gcnew System::Windows::Forms::TextBox());

this->textBox2\_Password = (gcnew System::Windows::Forms::TextBox());

this->ErrorLogin = (gcnew System::Windows::Forms::Label());

this->ErrorPassword = (gcnew System::Windows::Forms::Label());

this->radioButton1 = (gcnew System::Windows::Forms::RadioButton());

this->radioButton2 = (gcnew System::Windows::Forms::RadioButton());

this->panel1 = (gcnew System::Windows::Forms::Panel());

this->CreateAccount = (gcnew System::Windows::Forms::Button());

this->label5 = (gcnew System::Windows::Forms::Label());

this->label6 = (gcnew System::Windows::Forms::Label());

this->panel1->SuspendLayout();

this->SuspendLayout();

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(57, 9);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(167, 13);

this->label1->TabIndex = 0;

this->label1->Text = L"Введите данные пользователя:";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(7, 41);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(41, 13);

this->label2->TabIndex = 1;

this->label2->Text = L"Логин:";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(7, 80);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(48, 13);

this->label3->TabIndex = 1;

this->label3->Text = L"Пароль:";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(6, 130);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(78, 13);

this->label4->TabIndex = 1;

this->label4->Text = L"Тип аккаунта:";

//

// textBox1\_Login

//

this->textBox1\_Login->Location = System::Drawing::Point(93, 38);

this->textBox1\_Login->Name = L"textBox1\_Login";

this->textBox1\_Login->Size = System::Drawing::Size(162, 20);

this->textBox1\_Login->TabIndex = 2;

//

// textBox2\_Password

//

this->textBox2\_Password->Location = System::Drawing::Point(93, 77);

this->textBox2\_Password->Name = L"textBox2\_Password";

this->textBox2\_Password->Size = System::Drawing::Size(162, 20);

this->textBox2\_Password->TabIndex = 4;

this->textBox2\_Password->UseSystemPasswordChar = true;

//

// ErrorLogin

//

this->ErrorLogin->AutoSize = true;

this->ErrorLogin->ForeColor = System::Drawing::Color::Red;

this->ErrorLogin->Location = System::Drawing::Point(93, 61);

this->ErrorLogin->Name = L"ErrorLogin";

this->ErrorLogin->Size = System::Drawing::Size(0, 13);

this->ErrorLogin->TabIndex = 3;

//

// ErrorPassword

//

this->ErrorPassword->AutoSize = true;

this->ErrorPassword->ForeColor = System::Drawing::Color::Red;

this->ErrorPassword->Location = System::Drawing::Point(93, 100);

this->ErrorPassword->Name = L"ErrorPassword";

this->ErrorPassword->Size = System::Drawing::Size(0, 13);

this->ErrorPassword->TabIndex = 3;

//

// radioButton1

//

this->radioButton1->AutoSize = true;

this->radioButton1->Checked = true;

this->radioButton1->Location = System::Drawing::Point(5, 3);

this->radioButton1->Name = L"radioButton1";

this->radioButton1->Size = System::Drawing::Size(166, 17);

this->radioButton1->TabIndex = 6;

this->radioButton1->TabStop = true;

this->radioButton1->Text = L"Стандартный пользователь";

this->radioButton1->UseVisualStyleBackColor = true;

//

// radioButton2

//

this->radioButton2->AutoSize = true;

this->radioButton2->Location = System::Drawing::Point(5, 23);

this->radioButton2->Name = L"radioButton2";

this->radioButton2->Size = System::Drawing::Size(104, 17);

this->radioButton2->TabIndex = 8;

this->radioButton2->Text = L"Администратор";

this->radioButton2->UseVisualStyleBackColor = true;

//

// panel1

//

this->panel1->Controls->Add(this->radioButton1);

this->panel1->Controls->Add(this->radioButton2);

this->panel1->Location = System::Drawing::Point(92, 116);

this->panel1->Name = L"panel1";

this->panel1->Size = System::Drawing::Size(174, 43);

this->panel1->TabIndex = 5;

//

// CreateAccount

//

this->CreateAccount->Location = System::Drawing::Point(92, 165);

this->CreateAccount->Name = L"CreateAccount";

this->CreateAccount->Size = System::Drawing::Size(75, 23);

this->CreateAccount->TabIndex = 10;

this->CreateAccount->Text = L"Создать аккаунт";

this->CreateAccount->UseVisualStyleBackColor = true;

this->CreateAccount->Click += gcnew System::EventHandler(this, &CreateNewAccount::CreateAccount\_Click);

//

// label5

//

this->label5->AutoSize = true;

this->label5->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 8.25F, System::Drawing::FontStyle::Underline, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label5->ForeColor = System::Drawing::SystemColors::ControlDark;

this->label5->Location = System::Drawing::Point(13, 22);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(254, 13);

this->label5->TabIndex = 0;

this->label5->Text = L"(Наведите, чтобы получить больше информации)";

//

// label6

//

this->label6->AutoSize = true;

this->label6->Cursor = System::Windows::Forms::Cursors::Hand;

this->label6->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label6->ForeColor = System::Drawing::SystemColors::HotTrack;

this->label6->Location = System::Drawing::Point(6, 93);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(57, 13);

this->label6->TabIndex = 11;

this->label6->Text = L"(показать)";

this->label6->MouseDown += gcnew System::Windows::Forms::MouseEventHandler(this, &CreateNewAccount::label6\_MouseDown);

this->label6->MouseUp += gcnew System::Windows::Forms::MouseEventHandler(this, &CreateNewAccount::label6\_MouseUp);

//

// CreateNewAccount

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(278, 194);

this->Controls->Add(this->label6);

this->Controls->Add(this->CreateAccount);

this->Controls->Add(this->panel1);

this->Controls->Add(this->ErrorPassword);

this->Controls->Add(this->ErrorLogin);

this->Controls->Add(this->textBox2\_Password);

this->Controls->Add(this->textBox1\_Login);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->label5);

this->Controls->Add(this->label1);

this->Enabled = false;

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedDialog;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"CreateNewAccount";

this->Text = L"Создать новый аккаунт";

this->panel1->ResumeLayout(false);

this->panel1->PerformLayout();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void CreateAccount\_Click(System::Object^ sender, System::EventArgs^ e) {

array <Char>^ key = { 1,2,3,4,5,6,7,8,9 };

array <Char>^ login = textBox1\_Login->Text->ToCharArray();

array <Char>^ password = textBox2\_Password->Text->ToCharArray();

array <Char>^ typeOfAcc = { ((radioButton1->Checked) ? 'U' : 'A') };

int isCorrect = 1;

do {

if ((login->Length < 2) || (login->Length > 8)) {

ErrorLogin->Text = "\*Слишком короткий/длинный логин";

isCorrect = 0;

break;

}

for (int i = 0; i < login->Length; i++) {

if (!((login[i] >= '0' && login[i] <= '9') || (login[i] <= 'Z' && login[i] >= 'A') || (login[i] <= 'z' && login[i] >= 'a'))) {

ErrorLogin->Text = "\*Логин содержит недопустимые символы";

isCorrect = 0;

break;

}

}

if (!isCorrect) break;

else {

if ((password->Length < 2) || (password->Length > 8)) {

ErrorPassword->Text = "\*Слишком короткий/длинный пароль";

isCorrect = 0;

break;

}

for (int i = 0; i < password->Length; i++) {

if (!((password[i] >= '0' && password[i] <= '9') || (password[i] <= 'Z' && password[i] >= 'A') || (password[i] <= 'z' && password[i] >= 'a'))) {

ErrorPassword->Text = "\*Пароль содержит недопустимые символы";

isCorrect = 0;

break;

}

}

}

StreamReader ^checking = gcnew StreamReader("AccessAccounts.txt");

String ^row;

for (int i = 0; i < login->Length; i++)

login[i] ^= key[i];

while (checking->Peek() >= 0) {

row = checking->ReadLine();

array <String^>^ cells = row->Split(' ', '\0');

array <Char>^ loginNew = cells[0]->ToCharArray();

Int32 isLoginAccepted = 1;

if (login->Length == loginNew->Length) {

for (int i = 0; i < login->Length; i++)

if (login[i] != loginNew[i]) {

isLoginAccepted = 0;

break;

}

}

if (!isLoginAccepted) {

ErrorLogin->Text = "Логин уже используется";

isCorrect = 0;

break;

}

}

checking->Close();

} while (false);

if (isCorrect) {

StreamWriter ^write = gcnew StreamWriter("AccessAccounts.txt", true);

for (int i = 0; i < login->Length; i++)

login[i] ^= key[i];

for (int i = 0; i < password->Length; i++)

password[i] ^= key[8 - i];

write->Write(login);

write->Write(" ");

write->Write(password);

write->Write(" ");

write->Write(typeOfAcc);

write->WriteLine();

write->Close();

this->DialogResult = System::Windows::Forms::DialogResult::OK;

}

}

private: System::Void label6\_MouseDown(System::Object^ sender, System::Windows::Forms::MouseEventArgs^ e) {

textBox2\_Password->UseSystemPasswordChar = false;

}

private: System::Void label6\_MouseUp(System::Object^ sender, System::Windows::Forms::MouseEventArgs^ e) {

textBox2\_Password->UseSystemPasswordChar = true;

}

};

}

AddingForm.h

#pragma once

namespace Base {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для AddingForm

/// </summary>

public ref class AddingForm : public System::Windows::Forms::Form

{

public:

AddingForm()

{

InitializeComponent();

}

AddingForm(DataGridView ^old)

{

InitializeComponent();

last = old;

}

DataGridView ^last;

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~AddingForm()

{

if (components)

{

delete components;

}

}

private: bool Changing\_FullName() {

if (!(FullName->Text->CompareTo(""))) {

AddNew->Enabled = false;

if (!Error->Text->CompareTo(""))

Error->Text = "Поле \"ФИО\" не может быть пустым!";

return false;

}

else {

AddNew->Enabled = true;

Error->Text = "";

return true;

}

}

private: bool Changing\_Number() {

if (maskedTextBox1\_Phone->Text->Length != 10) {

if (!Error->Text->CompareTo(""))

Error->Text = "В поле \"Телефонный номер\" номер введен не полностью!";

AddNew->Enabled = false;

return false;

}

else {

Error->Text = "";

AddNew->Enabled = true;

return true;

}

}

private: bool Changing\_Year() {

if (maskedTextBox2\_Year->Text->CompareTo("")) {

if ((Int32::Parse(maskedTextBox2\_Year->Text) > DateTime::Now.Year) ||

(Int32::Parse(maskedTextBox2\_Year->Text) < (DateTime::Now.Year - 150))) {

if (!Error->Text->CompareTo(""))

Error->Text = "В поле \"Год\" год введен неверно!";

AddNew->Enabled = false;

return false;

}

else {

Error->Text = "";

AddNew->Enabled = true;

return true;

}

}

else {

if (!Error->Text->CompareTo(""))

Error->Text = "В поле \"Год\" год введен неверно!";

AddNew->Enabled = false;

return false;

}

}

private: bool Changing\_Type() {

if (!comboBox1->Text->CompareTo("")) {

if (!Error->Text->CompareTo(""))

Error->Text = "Тип оплаты не выбран!";

AddNew->Enabled = false;

return false;

}

else {

Error->Text = "";

AddNew->Enabled = true;

return true;

}

}

private: bool Changing\_Address() {

if (!Address->Text->CompareTo("")) {

if (!Error->Text->CompareTo(""))

Error->Text = "Поле \"Адрес\" не может быть пустым";

AddNew->Enabled = false;

return false;

}

else {

Error->Text = "";

AddNew->Enabled = true;

return true;

}

}

private: System::Windows::Forms::Label^ Info;

private: System::Windows::Forms::Button^ AddNew;

private: System::Windows::Forms::ComboBox^ comboBox1;

private: System::Windows::Forms::MaskedTextBox^ maskedTextBox2\_Year;

private: System::Windows::Forms::MaskedTextBox^ maskedTextBox1\_Phone;

private: System::Windows::Forms::TextBox^ Address;

private: System::Windows::Forms::TextBox^ FullName;

private: System::Windows::Forms::Label^ label6;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ Error;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(AddingForm::typeid));

this->Info = (gcnew System::Windows::Forms::Label());

this->AddNew = (gcnew System::Windows::Forms::Button());

this->comboBox1 = (gcnew System::Windows::Forms::ComboBox());

this->maskedTextBox2\_Year = (gcnew System::Windows::Forms::MaskedTextBox());

this->maskedTextBox1\_Phone = (gcnew System::Windows::Forms::MaskedTextBox());

this->Address = (gcnew System::Windows::Forms::TextBox());

this->FullName = (gcnew System::Windows::Forms::TextBox());

this->label6 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label5 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label1 = (gcnew System::Windows::Forms::Label());

this->Error = (gcnew System::Windows::Forms::Label());

this->SuspendLayout();

//

// Info

//

this->Info->AutoSize = true;

this->Info->Location = System::Drawing::Point(6, 4);

this->Info->Name = L"Info";

this->Info->Size = System::Drawing::Size(151, 13);

this->Info->TabIndex = 0;

this->Info->Text = L"Введите параметры записи:";

//

// AddNew

//

this->AddNew->Location = System::Drawing::Point(490, 68);

this->AddNew->Name = L"AddNew";

this->AddNew->Size = System::Drawing::Size(120, 23);

this->AddNew->TabIndex = 5;

this->AddNew->Text = L"Добавить запись";

this->AddNew->UseVisualStyleBackColor = true;

this->AddNew->Click += gcnew System::EventHandler(this, &AddingForm::AddNew\_Click);

//

// comboBox1

//

this->comboBox1->DropDownStyle = System::Windows::Forms::ComboBoxStyle::DropDownList;

this->comboBox1->FormattingEnabled = true;

this->comboBox1->Items->AddRange(gcnew cli::array< System::Object^ >(2) { L"NAL", L"CAR" });

this->comboBox1->Location = System::Drawing::Point(372, 42);

this->comboBox1->Name = L"comboBox1";

this->comboBox1->Size = System::Drawing::Size(57, 21);

this->comboBox1->TabIndex = 3;

this->comboBox1->TextChanged += gcnew System::EventHandler(this, &AddingForm::comboBox1\_TextChanged);

//

// maskedTextBox2\_Year

//

this->maskedTextBox2\_Year->Location = System::Drawing::Point(309, 42);

this->maskedTextBox2\_Year->Mask = L"9999";

this->maskedTextBox2\_Year->Name = L"maskedTextBox2\_Year";

this->maskedTextBox2\_Year->Size = System::Drawing::Size(57, 20);

this->maskedTextBox2\_Year->TabIndex = 2;

this->maskedTextBox2\_Year->ValidatingType = System::DateTime::typeid;

this->maskedTextBox2\_Year->TextChanged += gcnew System::EventHandler(this, &AddingForm::maskedTextBox2\_Year\_TextChanged);

//

// maskedTextBox1\_Phone

//

this->maskedTextBox1\_Phone->Location = System::Drawing::Point(190, 42);

this->maskedTextBox1\_Phone->Mask = L"(99)-99-99";

this->maskedTextBox1\_Phone->Name = L"maskedTextBox1\_Phone";

this->maskedTextBox1\_Phone->Size = System::Drawing::Size(113, 20);

this->maskedTextBox1\_Phone->TabIndex = 1;

this->maskedTextBox1\_Phone->TextChanged += gcnew System::EventHandler(this, &AddingForm::maskedTextBox1\_Phone\_TextChanged);

//

// Address

//

this->Address->Location = System::Drawing::Point(435, 42);

this->Address->Name = L"Address";

this->Address->Size = System::Drawing::Size(175, 20);

this->Address->TabIndex = 4;

this->Address->TextChanged += gcnew System::EventHandler(this, &AddingForm::Address\_TextChanged);

//

// FullName

//

this->FullName->Location = System::Drawing::Point(9, 42);

this->FullName->Name = L"FullName";

this->FullName->Size = System::Drawing::Size(175, 20);

this->FullName->TabIndex = 0;

this->FullName->TextChanged += gcnew System::EventHandler(this, &AddingForm::FullName\_TextChanged);

//

// label6

//

this->label6->AutoSize = true;

this->label6->Location = System::Drawing::Point(432, 26);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(95, 13);

this->label6->TabIndex = 16;

this->label6->Text = L"Домашний адрес";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(369, 26);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(66, 13);

this->label4->TabIndex = 17;

this->label4->Text = L"Тип оплаты";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Location = System::Drawing::Point(306, 26);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(25, 13);

this->label5->TabIndex = 18;

this->label5->Text = L"Год";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(306, 26);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(25, 13);

this->label3->TabIndex = 19;

this->label3->Text = L"Год";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(187, 26);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(107, 13);

this->label2->TabIndex = 20;

this->label2->Text = L"Телефонный номер";

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(6, 26);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(34, 13);

this->label1->TabIndex = 21;

this->label1->Text = L"ФИО";

//

// Error

//

this->Error->AutoSize = true;

this->Error->ForeColor = System::Drawing::Color::Red;

this->Error->Location = System::Drawing::Point(6, 73);

this->Error->Name = L"Error";

this->Error->Size = System::Drawing::Size(0, 13);

this->Error->TabIndex = 27;

//

// AddingForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(619, 101);

this->Controls->Add(this->Error);

this->Controls->Add(this->comboBox1);

this->Controls->Add(this->maskedTextBox2\_Year);

this->Controls->Add(this->maskedTextBox1\_Phone);

this->Controls->Add(this->Address);

this->Controls->Add(this->FullName);

this->Controls->Add(this->label6);

this->Controls->Add(this->label4);

this->Controls->Add(this->label5);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->AddNew);

this->Controls->Add(this->Info);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedDialog;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"AddingForm";

this->Text = L"Добавление новой записи";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void AddNew\_Click(System::Object^ sender, System::EventArgs^ e) {

if (Changing\_FullName() && Changing\_Number() && Changing\_Year() && Changing\_Type() && Changing\_Address())

last->Rows->Add((FullName->Text + " " + maskedTextBox1\_Phone->Text + " " + maskedTextBox2\_Year->Text + " " + comboBox1->SelectedItem->ToString() + " " + Address->Text + "\n")->Split(' '));

}

private: System::Void FullName\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_FullName();

}

private: System::Void maskedTextBox1\_Phone\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Number();

}

private: System::Void maskedTextBox2\_Year\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Year();

}

private: System::Void comboBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Type();

}

private: System::Void Address\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Address();

}

};

EditingForm.h

#pragma once

namespace Base {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для EditingForm

/// </summary>

public ref class EditingForm : public System::Windows::Forms::Form

{

public:

EditingForm(DataGridView ^last)

{

InitializeComponent();

old = last;

FullName->Text = (old->Rows[old->CurrentRow->Index]->Cells[0]->Value)->ToString();

maskedTextBox1\_Phone->Text = (old->Rows[old->CurrentRow->Index]->Cells[1]->Value)->ToString();

maskedTextBox2\_Year->Text = (old->Rows[old->CurrentRow->Index]->Cells[2]->Value)->ToString();

comboBox1->Text = (old->Rows[old->CurrentRow->Index]->Cells[3]->Value)->ToString();

Address->Text = (old->Rows[old->CurrentRow->Index]->Cells[4]->Value)->ToString();

}

EditingForm() {

InitializeComponent();

}

DataGridView ^old; // Указатель на datagridView из главной формы

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~EditingForm()

{

if (components)

{

delete components;

}

}

private: bool Changing\_FullName() {

if (!(FullName->Text->CompareTo(""))) {

Edit->Enabled = false;

if (!Error->Text->CompareTo(""))

Error->Text = "Поле \"ФИО\" не может быть пустым!";

return false;

}

else {

Edit->Enabled = true;

Error->Text = "";

return true;

}

}

private: bool Changing\_Number() {

if (maskedTextBox1\_Phone->Text->Length != 10) {

if (!Error->Text->CompareTo(""))

Error->Text = "В поле \"Телефонный номер\" номер введен не полностью!";

Edit->Enabled = false;

return false;

}

else {

Error->Text = "";

Edit->Enabled = true;

return true;

}

}

private: bool Changing\_Year() {

if (maskedTextBox2\_Year->Text->CompareTo("")) {

if ((Int32::Parse(maskedTextBox2\_Year->Text) > DateTime::Now.Year) ||

(Int32::Parse(maskedTextBox2\_Year->Text) < (DateTime::Now.Year - 150))) {

if (!Error->Text->CompareTo(""))

Error->Text = "В поле \"Год\" год введен неверно!";

Edit->Enabled = false;

return false;

}

else {

Error->Text = "";

Edit->Enabled = true;

return true;

}

}

else {

if (!Error->Text->CompareTo(""))

Error->Text = "В поле \"Год\" год введен неверно!";

Edit->Enabled = false;

return false;

}

}

private: bool Changing\_Type() {

if (!comboBox1->Text->CompareTo("")) {

if (!Error->Text->CompareTo(""))

Error->Text = "Тип оплаты не выбран!";

Edit->Enabled = false;

return false;

}

else {

Error->Text = "";

Edit->Enabled = true;

return true;

}

}

private: bool Changing\_Address() {

if (!Address->Text->CompareTo("")) {

if (!Error->Text->CompareTo(""))

Error->Text = "Поле \"Адрес\" не может быть пустым";

Edit->Enabled = false;

return false;

}

else {

Error->Text = "";

Edit->Enabled = true;

return true;

}

}

private: System::Windows::Forms::ComboBox^ comboBox1;

private: System::Windows::Forms::MaskedTextBox^ maskedTextBox2\_Year;

private: System::Windows::Forms::MaskedTextBox^ maskedTextBox1\_Phone;

private: System::Windows::Forms::TextBox^ Address;

private: System::Windows::Forms::TextBox^ FullName;

private: System::Windows::Forms::Label^ label6;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Button^ Edit;

private: System::Windows::Forms::Label^ Info;

private: System::Windows::Forms::Label^ Error;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(EditingForm::typeid));

this->comboBox1 = (gcnew System::Windows::Forms::ComboBox());

this->maskedTextBox2\_Year = (gcnew System::Windows::Forms::MaskedTextBox());

this->maskedTextBox1\_Phone = (gcnew System::Windows::Forms::MaskedTextBox());

this->Address = (gcnew System::Windows::Forms::TextBox());

this->FullName = (gcnew System::Windows::Forms::TextBox());

this->label6 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label5 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label1 = (gcnew System::Windows::Forms::Label());

this->Edit = (gcnew System::Windows::Forms::Button());

this->Info = (gcnew System::Windows::Forms::Label());

this->Error = (gcnew System::Windows::Forms::Label());

this->SuspendLayout();

//

// comboBox1

//

this->comboBox1->DropDownStyle = System::Windows::Forms::ComboBoxStyle::DropDownList;

this->comboBox1->FormattingEnabled = true;

this->comboBox1->Items->AddRange(gcnew cli::array< System::Object^ >(2) { L"NAL", L"CAR" });

this->comboBox1->Location = System::Drawing::Point(375, 49);

this->comboBox1->Name = L"comboBox1";

this->comboBox1->Size = System::Drawing::Size(57, 21);

this->comboBox1->TabIndex = 3;

this->comboBox1->TextChanged += gcnew System::EventHandler(this, &EditingForm::comboBox1\_TextChanged);

//

// maskedTextBox2\_Year

//

this->maskedTextBox2\_Year->Location = System::Drawing::Point(312, 49);

this->maskedTextBox2\_Year->Mask = L"9999";

this->maskedTextBox2\_Year->Name = L"maskedTextBox2\_Year";

this->maskedTextBox2\_Year->Size = System::Drawing::Size(57, 20);

this->maskedTextBox2\_Year->TabIndex = 2;

this->maskedTextBox2\_Year->ValidatingType = System::DateTime::typeid;

this->maskedTextBox2\_Year->TextChanged += gcnew System::EventHandler(this, &EditingForm::maskedTextBox2\_Year\_TextChanged);

//

// maskedTextBox1\_Phone

//

this->maskedTextBox1\_Phone->Location = System::Drawing::Point(193, 49);

this->maskedTextBox1\_Phone->Mask = L"(99)-99-99";

this->maskedTextBox1\_Phone->Name = L"maskedTextBox1\_Phone";

this->maskedTextBox1\_Phone->Size = System::Drawing::Size(113, 20);

this->maskedTextBox1\_Phone->TabIndex = 1;

this->maskedTextBox1\_Phone->TextChanged += gcnew System::EventHandler(this, &EditingForm::maskedTextBox1\_Phone\_TextChanged);

//

// Address

//

this->Address->Location = System::Drawing::Point(438, 49);

this->Address->Name = L"Address";

this->Address->Size = System::Drawing::Size(175, 20);

this->Address->TabIndex = 4;

this->Address->TextChanged += gcnew System::EventHandler(this, &EditingForm::Address\_TextChanged);

//

// FullName

//

this->FullName->Location = System::Drawing::Point(12, 49);

this->FullName->Name = L"FullName";

this->FullName->Size = System::Drawing::Size(175, 20);

this->FullName->TabIndex = 0;

this->FullName->TextChanged += gcnew System::EventHandler(this, &EditingForm::FullName\_TextChanged);

//

// label6

//

this->label6->AutoSize = true;

this->label6->Location = System::Drawing::Point(435, 33);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(95, 13);

this->label6->TabIndex = 29;

this->label6->Text = L"Домашний адрес";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(372, 33);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(66, 13);

this->label4->TabIndex = 30;

this->label4->Text = L"Тип оплаты";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Location = System::Drawing::Point(309, 33);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(25, 13);

this->label5->TabIndex = 31;

this->label5->Text = L"Год";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(309, 33);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(25, 13);

this->label3->TabIndex = 32;

this->label3->Text = L"Год";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(190, 33);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(107, 13);

this->label2->TabIndex = 33;

this->label2->Text = L"Телефонный номер";

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(9, 33);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(34, 13);

this->label1->TabIndex = 34;

this->label1->Text = L"ФИО";

//

// Edit

//

this->Edit->Location = System::Drawing::Point(493, 75);

this->Edit->Name = L"Edit";

this->Edit->Size = System::Drawing::Size(120, 23);

this->Edit->TabIndex = 5;

this->Edit->Text = L"Изменить";

this->Edit->UseVisualStyleBackColor = true;

this->Edit->Click += gcnew System::EventHandler(this, &EditingForm::AddNew\_Click);

//

// Info

//

this->Info->AutoSize = true;

this->Info->Location = System::Drawing::Point(9, 11);

this->Info->Name = L"Info";

this->Info->Size = System::Drawing::Size(151, 13);

this->Info->TabIndex = 27;

this->Info->Text = L"Введите параметры записи:";

//

// Error

//

this->Error->AutoSize = true;

this->Error->ForeColor = System::Drawing::Color::Red;

this->Error->Location = System::Drawing::Point(12, 93);

this->Error->Name = L"Error";

this->Error->Size = System::Drawing::Size(0, 13);

this->Error->TabIndex = 40;

//

// EditingForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(628, 109);

this->Controls->Add(this->Error);

this->Controls->Add(this->comboBox1);

this->Controls->Add(this->maskedTextBox2\_Year);

this->Controls->Add(this->maskedTextBox1\_Phone);

this->Controls->Add(this->Address);

this->Controls->Add(this->FullName);

this->Controls->Add(this->label6);

this->Controls->Add(this->label4);

this->Controls->Add(this->label5);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->Edit);

this->Controls->Add(this->Info);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedDialog;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"EditingForm";

this->Text = L"Изменение записи";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void AddNew\_Click(System::Object^ sender, System::EventArgs^ e) {

if (Changing\_FullName() && Changing\_Number() && Changing\_Year() && Changing\_Type() && Changing\_Address()) {

old->Rows[old->CurrentRow->Index]->Cells[0]->Value = FullName->Text;

old->Rows[old->CurrentRow->Index]->Cells[1]->Value = maskedTextBox1\_Phone->Text;

old->Rows[old->CurrentRow->Index]->Cells[2]->Value = maskedTextBox2\_Year->Text;

old->Rows[old->CurrentRow->Index]->Cells[3]->Value = comboBox1->Text;

old->Rows[old->CurrentRow->Index]->Cells[4]->Value = Address->Text;

}

}

private: System::Void FullName\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_FullName();

}

private: System::Void maskedTextBox1\_Phone\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Number();

}

private: System::Void maskedTextBox2\_Year\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Year();

}

private: System::Void comboBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Type();

}

private: System::Void Address\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Address();

}

};

}

Request.h

#pragma once

namespace Base {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

/// <summary>

/// Сводка для Request

/// </summary>

public ref class Request : public System::Windows::Forms::Form

{

public:

Request(String ^NameOfFile\_Old)

{

InitializeComponent();

this->nameOfFile = NameOfFile\_Old;

StreamReader ^tempFile = gcnew StreamReader(nameOfFile);

while (tempFile->Peek() >= 0) {

array <String^>^ cells = (tempFile->ReadLine())->Split(' ','\0');

if(cells[0]->CompareTo(""))

dataGridView1->Rows->Add(cells);

}

}

Request() {

InitializeComponent();

}

String ^nameOfFile;

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~Request()

{

if (components)

{

delete components;

}

}

private: Void Changing\_FullName() {

if(checkBox1\_FIO->Checked)

if (!(FullName->Text->CompareTo(""))) {

CreateRequest->Enabled = false;

if (!Error->Text->CompareTo(""))

Error->Text = "Поле \"ФИО\" не может быть пустым!";

}

else {

CreateRequest->Enabled = true;

Error->Text = "";

}

}

private: Void Changing\_Number() {

if(checkBox2\_Phone->Checked)

if (maskedTextBox1\_Phone->Text->Length != 10) {

if (!Error->Text->CompareTo(""))

Error->Text = "В поле \"Телефонный номер\" номер введен не полностью!";

CreateRequest->Enabled = false;

}

else {

Error->Text = "";

CreateRequest->Enabled = true;

}

}

private: Void Changing\_Year() {

if (checkBox3\_Year->Checked) {

if (maskedTextBox2\_Year->Text->CompareTo("")) {

if ((Int32::Parse(maskedTextBox2\_Year->Text) > DateTime::Now.Year) ||

(Int32::Parse(maskedTextBox2\_Year->Text) < (DateTime::Now.Year - 150))) {

if (!Error->Text->CompareTo(""))

Error->Text = "В поле \"Год\" год введен неверно!";

CreateRequest->Enabled = false;

}

else {

Error->Text = "";

CreateRequest->Enabled = true;

}

}

else {

if (!Error->Text->CompareTo(""))

Error->Text = "В поле \"Год\" год введен неверно!";

CreateRequest->Enabled = false;

}

}

}

private: Void Changing\_Type() {

if(checkBox4\_Type->Checked)

if (!comboBox1->Text->CompareTo("")) {

if (!Error->Text->CompareTo(""))

Error->Text = "Тип оплаты не выбран!";

CreateRequest->Enabled = false;

}

else {

Error->Text = "";

CreateRequest->Enabled = true;

}

}

private: Void Changing\_Address(){

if(checkBox5\_Address->Checked)

if (!Address->Text->CompareTo("")) {

if (!Error->Text->CompareTo(""))

Error->Text = "Поле \"Адрес\" не может быть пустым";

CreateRequest->Enabled = false;

}

else {

Error->Text = "";

CreateRequest->Enabled = true;

}

}

private: System::Windows::Forms::Label^ Rules;

private: System::Windows::Forms::TextBox^ FullName;

private: System::Windows::Forms::TextBox^ Address;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Label^ label6;

private: System::Windows::Forms::Label^ Errors;

private: System::Windows::Forms::Button^ CreateRequest;

private: System::Windows::Forms::DataGridView^ dataGridView1;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ fio;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ PhoneNumber;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ dataGridViewTextBoxColumn1;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ TypeOfPaying;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Adress;

private: System::Windows::Forms::Button^ ResetRequest;

private: System::Windows::Forms::CheckBox^ checkBox1\_FIO;

private: System::Windows::Forms::CheckBox^ checkBox2\_Phone;

private: System::Windows::Forms::CheckBox^ checkBox3\_Year;

private: System::Windows::Forms::CheckBox^ checkBox4\_Type;

private: System::Windows::Forms::CheckBox^ checkBox5\_Address;

private: System::Windows::Forms::MaskedTextBox^ maskedTextBox1\_Phone;

private: System::Windows::Forms::MaskedTextBox^ maskedTextBox2\_Year;

private: System::Windows::Forms::ComboBox^ comboBox1;

private: System::Windows::Forms::Label^ Error;

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(Request::typeid));

this->Rules = (gcnew System::Windows::Forms::Label());

this->FullName = (gcnew System::Windows::Forms::TextBox());

this->Address = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label5 = (gcnew System::Windows::Forms::Label());

this->label6 = (gcnew System::Windows::Forms::Label());

this->Errors = (gcnew System::Windows::Forms::Label());

this->CreateRequest = (gcnew System::Windows::Forms::Button());

this->dataGridView1 = (gcnew System::Windows::Forms::DataGridView());

this->fio = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->PhoneNumber = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->dataGridViewTextBoxColumn1 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->TypeOfPaying = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Adress = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->ResetRequest = (gcnew System::Windows::Forms::Button());

this->checkBox1\_FIO = (gcnew System::Windows::Forms::CheckBox());

this->checkBox2\_Phone = (gcnew System::Windows::Forms::CheckBox());

this->checkBox3\_Year = (gcnew System::Windows::Forms::CheckBox());

this->checkBox4\_Type = (gcnew System::Windows::Forms::CheckBox());

this->checkBox5\_Address = (gcnew System::Windows::Forms::CheckBox());

this->maskedTextBox1\_Phone = (gcnew System::Windows::Forms::MaskedTextBox());

this->maskedTextBox2\_Year = (gcnew System::Windows::Forms::MaskedTextBox());

this->comboBox1 = (gcnew System::Windows::Forms::ComboBox());

this->Error = (gcnew System::Windows::Forms::Label());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->BeginInit();

this->SuspendLayout();

//

// Rules

//

this->Rules->AutoSize = true;

this->Rules->Location = System::Drawing::Point(1, 9);

this->Rules->Name = L"Rules";

this->Rules->Size = System::Drawing::Size(180, 13);

this->Rules->TabIndex = 0;

this->Rules->Text = L"Введите запрашиваемые данные:";

//

// FullName

//

this->FullName->Enabled = false;

this->FullName->Location = System::Drawing::Point(8, 44);

this->FullName->Name = L"FullName";

this->FullName->Size = System::Drawing::Size(175, 20);

this->FullName->TabIndex = 0;

this->FullName->TextChanged += gcnew System::EventHandler(this, &Request::FullName\_TextChanged);

//

// Address

//

this->Address->Enabled = false;

this->Address->Location = System::Drawing::Point(434, 44);

this->Address->Name = L"Address";

this->Address->Size = System::Drawing::Size(175, 20);

this->Address->TabIndex = 4;

this->Address->TextChanged += gcnew System::EventHandler(this, &Request::Address\_TextChanged);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(5, 28);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(34, 13);

this->label1->TabIndex = 0;

this->label1->Text = L"ФИО";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(186, 28);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(107, 13);

this->label2->TabIndex = 0;

this->label2->Text = L"Телефонный номер";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(305, 28);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(25, 13);

this->label3->TabIndex = 0;

this->label3->Text = L"Год";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(368, 28);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(66, 13);

this->label4->TabIndex = 0;

this->label4->Text = L"Тип оплаты";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Location = System::Drawing::Point(305, 28);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(25, 13);

this->label5->TabIndex = 0;

this->label5->Text = L"Год";

//

// label6

//

this->label6->AutoSize = true;

this->label6->Location = System::Drawing::Point(431, 28);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(95, 13);

this->label6->TabIndex = 0;

this->label6->Text = L"Домашний адрес";

//

// Errors

//

this->Errors->AutoSize = true;

this->Errors->ForeColor = System::Drawing::SystemColors::HotTrack;

this->Errors->Location = System::Drawing::Point(12, 119);

this->Errors->Name = L"Errors";

this->Errors->Size = System::Drawing::Size(302, 13);

this->Errors->TabIndex = 0;

this->Errors->Text = L"\* Оставьте поле пустым, если оно не участвует в запросе";

//

// CreateRequest

//

this->CreateRequest->Enabled = false;

this->CreateRequest->Location = System::Drawing::Point(487, 88);

this->CreateRequest->Name = L"CreateRequest";

this->CreateRequest->Size = System::Drawing::Size(120, 23);

this->CreateRequest->TabIndex = 5;

this->CreateRequest->Text = L"Выполнить запрос";

this->CreateRequest->UseVisualStyleBackColor = true;

this->CreateRequest->Click += gcnew System::EventHandler(this, &Request::CreateRequest\_Click);

//

// dataGridView1

//

this->dataGridView1->AllowUserToOrderColumns = true;

this->dataGridView1->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->dataGridView1->Columns->AddRange(gcnew cli::array< System::Windows::Forms::DataGridViewColumn^ >(5) {

this->fio, this->PhoneNumber,

this->dataGridViewTextBoxColumn1, this->TypeOfPaying, this->Adress

});

this->dataGridView1->Location = System::Drawing::Point(15, 135);

this->dataGridView1->Name = L"dataGridView1";

this->dataGridView1->ReadOnly = true;

this->dataGridView1->Size = System::Drawing::Size(595, 329);

this->dataGridView1->TabIndex = 3;

//

// fio

//

this->fio->HeaderText = L"ФИО";

this->fio->Name = L"fio";

this->fio->ReadOnly = true;

this->fio->Width = 150;

//

// PhoneNumber

//

this->PhoneNumber->HeaderText = L"Номер телефона";

this->PhoneNumber->Name = L"PhoneNumber";

this->PhoneNumber->ReadOnly = true;

//

// dataGridViewTextBoxColumn1

//

this->dataGridViewTextBoxColumn1->HeaderText = L"Год";

this->dataGridViewTextBoxColumn1->Name = L"dataGridViewTextBoxColumn1";

this->dataGridViewTextBoxColumn1->ReadOnly = true;

this->dataGridViewTextBoxColumn1->Width = 50;

//

// TypeOfPaying

//

this->TypeOfPaying->HeaderText = L"Тип оплаты";

this->TypeOfPaying->Name = L"TypeOfPaying";

this->TypeOfPaying->ReadOnly = true;

this->TypeOfPaying->Width = 50;

//

// Adress

//

this->Adress->HeaderText = L"Адрес";

this->Adress->Name = L"Adress";

this->Adress->ReadOnly = true;

this->Adress->Width = 200;

//

// ResetRequest

//

this->ResetRequest->Location = System::Drawing::Point(361, 88);

this->ResetRequest->Name = L"ResetRequest";

this->ResetRequest->Size = System::Drawing::Size(120, 23);

this->ResetRequest->TabIndex = 11;

this->ResetRequest->Text = L"Сбросить запрос";

this->ResetRequest->UseVisualStyleBackColor = true;

this->ResetRequest->Click += gcnew System::EventHandler(this, &Request::ResetRequest\_Click);

//

// checkBox1\_FIO

//

this->checkBox1\_FIO->AutoSize = true;

this->checkBox1\_FIO->Location = System::Drawing::Point(88, 71);

this->checkBox1\_FIO->Name = L"checkBox1\_FIO";

this->checkBox1\_FIO->Size = System::Drawing::Size(15, 14);

this->checkBox1\_FIO->TabIndex = 6;

this->checkBox1\_FIO->UseVisualStyleBackColor = true;

this->checkBox1\_FIO->CheckedChanged += gcnew System::EventHandler(this, &Request::checkBox1\_FIO\_CheckedChanged);

//

// checkBox2\_Phone

//

this->checkBox2\_Phone->AutoSize = true;

this->checkBox2\_Phone->Location = System::Drawing::Point(241, 70);

this->checkBox2\_Phone->Name = L"checkBox2\_Phone";

this->checkBox2\_Phone->Size = System::Drawing::Size(15, 14);

this->checkBox2\_Phone->TabIndex = 7;

this->checkBox2\_Phone->UseVisualStyleBackColor = true;

this->checkBox2\_Phone->CheckedChanged += gcnew System::EventHandler(this, &Request::checkBox2\_Phone\_CheckedChanged);

//

// checkBox3\_Year

//

this->checkBox3\_Year->AutoSize = true;

this->checkBox3\_Year->Location = System::Drawing::Point(332, 70);

this->checkBox3\_Year->Name = L"checkBox3\_Year";

this->checkBox3\_Year->Size = System::Drawing::Size(15, 14);

this->checkBox3\_Year->TabIndex = 8;

this->checkBox3\_Year->UseVisualStyleBackColor = true;

this->checkBox3\_Year->CheckedChanged += gcnew System::EventHandler(this, &Request::checkBox3\_Year\_CheckedChanged);

//

// checkBox4\_Type

//

this->checkBox4\_Type->AutoSize = true;

this->checkBox4\_Type->Location = System::Drawing::Point(393, 70);

this->checkBox4\_Type->Name = L"checkBox4\_Type";

this->checkBox4\_Type->Size = System::Drawing::Size(15, 14);

this->checkBox4\_Type->TabIndex = 9;

this->checkBox4\_Type->UseVisualStyleBackColor = true;

this->checkBox4\_Type->CheckedChanged += gcnew System::EventHandler(this, &Request::checkBox4\_Type\_CheckedChanged);

//

// checkBox5\_Address

//

this->checkBox5\_Address->AutoSize = true;

this->checkBox5\_Address->Location = System::Drawing::Point(513, 70);

this->checkBox5\_Address->Name = L"checkBox5\_Address";

this->checkBox5\_Address->Size = System::Drawing::Size(15, 14);

this->checkBox5\_Address->TabIndex = 10;

this->checkBox5\_Address->UseVisualStyleBackColor = true;

this->checkBox5\_Address->CheckedChanged += gcnew System::EventHandler(this, &Request::checkBox5\_Address\_CheckedChanged);

//

// maskedTextBox1\_Phone

//

this->maskedTextBox1\_Phone->Enabled = false;

this->maskedTextBox1\_Phone->Location = System::Drawing::Point(189, 44);

this->maskedTextBox1\_Phone->Mask = L"(99)-99-99";

this->maskedTextBox1\_Phone->Name = L"maskedTextBox1\_Phone";

this->maskedTextBox1\_Phone->Size = System::Drawing::Size(113, 20);

this->maskedTextBox1\_Phone->TabIndex = 1;

this->maskedTextBox1\_Phone->TextChanged += gcnew System::EventHandler(this, &Request::maskedTextBox1\_Phone\_TextChanged);

//

// maskedTextBox2\_Year

//

this->maskedTextBox2\_Year->Enabled = false;

this->maskedTextBox2\_Year->Location = System::Drawing::Point(308, 44);

this->maskedTextBox2\_Year->Mask = L"9999";

this->maskedTextBox2\_Year->Name = L"maskedTextBox2\_Year";

this->maskedTextBox2\_Year->Size = System::Drawing::Size(57, 20);

this->maskedTextBox2\_Year->TabIndex = 2;

this->maskedTextBox2\_Year->ValidatingType = System::DateTime::typeid;

this->maskedTextBox2\_Year->TextChanged += gcnew System::EventHandler(this, &Request::maskedTextBox2\_Year\_TextChanged);

//

// comboBox1

//

this->comboBox1->DropDownStyle = System::Windows::Forms::ComboBoxStyle::DropDownList;

this->comboBox1->FormattingEnabled = true;

this->comboBox1->Items->AddRange(gcnew cli::array< System::Object^ >(2) { L"NAL", L"CAR" });

this->comboBox1->Location = System::Drawing::Point(371, 44);

this->comboBox1->Name = L"comboBox1";

this->comboBox1->Size = System::Drawing::Size(57, 21);

this->comboBox1->TabIndex = 3;

this->comboBox1->TextChanged += gcnew System::EventHandler(this, &Request::comboBox1\_TextChanged);

//

// Error

//

this->Error->AutoSize = true;

this->Error->ForeColor = System::Drawing::Color::Red;

this->Error->Location = System::Drawing::Point(5, 98);

this->Error->Name = L"Error";

this->Error->Size = System::Drawing::Size(0, 13);

this->Error->TabIndex = 0;

//

// Request

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(618, 476);

this->Controls->Add(this->comboBox1);

this->Controls->Add(this->maskedTextBox2\_Year);

this->Controls->Add(this->maskedTextBox1\_Phone);

this->Controls->Add(this->checkBox5\_Address);

this->Controls->Add(this->checkBox4\_Type);

this->Controls->Add(this->checkBox3\_Year);

this->Controls->Add(this->checkBox2\_Phone);

this->Controls->Add(this->checkBox1\_FIO);

this->Controls->Add(this->dataGridView1);

this->Controls->Add(this->ResetRequest);

this->Controls->Add(this->CreateRequest);

this->Controls->Add(this->Address);

this->Controls->Add(this->FullName);

this->Controls->Add(this->label6);

this->Controls->Add(this->label4);

this->Controls->Add(this->label5);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->Error);

this->Controls->Add(this->Errors);

this->Controls->Add(this->Rules);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedDialog;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"Request";

this->Text = L"Запрос";

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void checkBox1\_FIO\_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {

if (checkBox1\_FIO->Checked) {

FullName->Enabled = true;

CreateRequest->Enabled = false;

}

else {

Error->Text = "";

Changing\_Number();

Changing\_Type();

Changing\_Year();

Changing\_Address();

FullName->Enabled = false;

FullName->Text = "";

if (!(checkBox1\_FIO->Checked || checkBox2\_Phone->Checked || checkBox3\_Year->Checked ||

checkBox4\_Type->Checked || checkBox5\_Address->Checked)) CreateRequest->Enabled = false;

}

}

private: System::Void checkBox2\_Phone\_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {

if (checkBox2\_Phone->Checked) {

maskedTextBox1\_Phone->Enabled = true;

CreateRequest->Enabled = false;

}

else {

Error->Text = "";

Changing\_FullName();

Changing\_Type();

Changing\_Year();

Changing\_Address();

maskedTextBox1\_Phone->Enabled = false;

maskedTextBox1\_Phone->Text = "";

if (!(checkBox1\_FIO->Checked || checkBox2\_Phone->Checked || checkBox3\_Year->Checked ||

checkBox4\_Type->Checked || checkBox5\_Address->Checked)) CreateRequest->Enabled = false;

}

}

private: System::Void checkBox3\_Year\_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {

if (checkBox3\_Year->Checked) {

maskedTextBox2\_Year->Enabled = true;

CreateRequest->Enabled = false;

}

else {

Error->Text = "";

Changing\_Number();

Changing\_Type();

Changing\_FullName();

Changing\_Address();

maskedTextBox2\_Year->Enabled = false;

maskedTextBox2\_Year->Text = "";

if (!(checkBox1\_FIO->Checked || checkBox2\_Phone->Checked || checkBox3\_Year->Checked ||

checkBox4\_Type->Checked || checkBox5\_Address->Checked)) CreateRequest->Enabled = false;

}

}

private: System::Void checkBox4\_Type\_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {

if (checkBox4\_Type->Checked) {

comboBox1->Enabled = true;

CreateRequest->Enabled = false;

}

else {

Error->Text = "";

Changing\_Number();

Changing\_FullName();

Changing\_Year();

Changing\_Address();

comboBox1->Enabled = false;

comboBox1->Text = "";

if (!(checkBox1\_FIO->Checked || checkBox2\_Phone->Checked || checkBox3\_Year->Checked ||

checkBox4\_Type->Checked || checkBox5\_Address->Checked)) CreateRequest->Enabled = false;

}

}

private: System::Void checkBox5\_Address\_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {

if (checkBox5\_Address->Checked) {

Address->Enabled = true;

CreateRequest->Enabled = false;

}

else {

Error->Text = "";

Changing\_Number();

Changing\_Type();

Changing\_Year();

Changing\_FullName();

Address->Enabled = false;

Address->Text = "";

if (!(checkBox1\_FIO->Checked || checkBox2\_Phone->Checked || checkBox3\_Year->Checked ||

checkBox4\_Type->Checked || checkBox5\_Address->Checked)) CreateRequest->Enabled = false;

}

}

private: System::Void CreateRequest\_Click(System::Object^ sender, System::EventArgs^ e) {

//Проверка корректности ввода

StreamReader ^tempFile = gcnew StreamReader(nameOfFile);

dataGridView1->Rows->Clear();

while (tempFile->Peek() >= 0) {

array <String^>^ cells = (tempFile->ReadLine())->Split(' ');

if (cells[0]->CompareTo("")) {

//Реализация запроса\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int isRight = 1;

if (checkBox1\_FIO->Checked) {

if (FullName->Text->CompareTo(cells[0])) {

isRight = 0;

continue;

}

}

if (checkBox2\_Phone->Checked) {

if (maskedTextBox1\_Phone->Text->CompareTo(cells[1])) {

isRight = 0;

continue;

}

}

if (checkBox3\_Year->Checked) {

if (maskedTextBox2\_Year->Text->CompareTo(cells[2])) {

isRight = 0;

continue;

}

}

if (checkBox4\_Type->Checked) { // Проблемы с combobox ом

if ((comboBox1->SelectedItem->ToString())->CompareTo(cells[3])) {

isRight = 0;

continue;

}

}

if (checkBox5\_Address->Checked) {

if (Address->Text->CompareTo(cells[4])) {

isRight = 0;

continue;

}

}

if (isRight) dataGridView1->Rows->Add(cells);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

}

}

}

private: System::Void ResetRequest\_Click(System::Object^ sender, System::EventArgs^ e) {

dataGridView1->Rows->Clear();

StreamReader ^tempFile = gcnew StreamReader(nameOfFile);

while (tempFile->Peek() >= 0) {

array <String^>^ cells = (tempFile->ReadLine())->Split(' ','\0');

if (cells[0]->CompareTo("")) {

dataGridView1->Rows->Add(cells);

}

}

}

private: System::Void FullName\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_FullName();

Changing\_Number();

Changing\_Type();

Changing\_Year();

Changing\_Address();

}

public: System::Void maskedTextBox1\_Phone\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Number();

Changing\_FullName();

Changing\_Type();

Changing\_Year();

Changing\_Address();

}

private: System::Void maskedTextBox2\_Year\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Year();

Changing\_Number();

Changing\_Type();

Changing\_FullName();

Changing\_Address();

}

private: System::Void comboBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Type();

Changing\_Number();

Changing\_FullName();

Changing\_Year();

Changing\_Address();

}

private: System::Void Address\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Changing\_Address();

Changing\_Number();

Changing\_Type();

Changing\_Year();

Changing\_FullName();

}

};

}

GameMetalSmith.h

#pragma once

#include <iostream>

#include <time.h>

namespace Base {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для GameMetalSmith

/// </summary>

public ref class GameMetalSmith : public System::Windows::Forms::Form

{

public:

GameMetalSmith(void)

{

InitializeComponent();

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~GameMetalSmith()

{

if (components)

{

delete components;

}

}

private: static array <Int32>^ last = gcnew array <Int32> (3);

private: static array <Int32>^ punches = gcnew array <Int32> { 20, 75, 160, 210, -20, -75, -160, -210 };

private: static Int32 streak = 0;

private: static Int32 round = 1;

public: static array <String^> ^punchesImages = gcnew array <String^>(8) {

"GameTextures\\Punch1.png",

"GameTextures\\Punch2.png",

"GameTextures\\Punch3.png",

"GameTextures\\Punch4.png",

"GameTextures\\Roll1.png",

"GameTextures\\Roll2.png",

"GameTextures\\Roll3.png",

"GameTextures\\Roll4.png"

};

protected:

private: System::Windows::Forms::TrackBar^ trackBar1;

private: System::Windows::Forms::ProgressBar^ progressBar1;

private: System::Windows::Forms::TrackBar^ trackBar2;

private: System::Windows::Forms::Button^ Punch1;

private: System::Windows::Forms::Button^ Punch2;

private: System::Windows::Forms::Button^ Roll1;

private: System::Windows::Forms::Button^ Punch3;

private: System::Windows::Forms::Button^ Punch4;

private: System::Windows::Forms::Button^ Roll3;

private: System::Windows::Forms::Panel^ panel1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Timer^ timer1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Panel^ panel5;

private: System::Windows::Forms::Button^ Roll2;

private: System::Windows::Forms::Button^ Roll4;

private: System::Windows::Forms::Panel^ panel3Ex;

private: System::Windows::Forms::Panel^ panel2Ex;

private: System::Windows::Forms::Panel^ panel1Ex;

private: System::Windows::Forms::Button^ NextLevel;

private: System::Windows::Forms::CheckBox^ checkBox3;

private: System::Windows::Forms::CheckBox^ checkBox2;

private: System::Windows::Forms::CheckBox^ checkBox1;

private: System::Windows::Forms::Panel^ panel3New;

private: System::Windows::Forms::Panel^ panel2New;

private: System::Windows::Forms::Panel^ panel1New;

private: System::Windows::Forms::Label^ label3;

private: System::ComponentModel::IContainer^ components;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->components = (gcnew System::ComponentModel::Container());

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(GameMetalSmith::typeid));

this->trackBar1 = (gcnew System::Windows::Forms::TrackBar());

this->progressBar1 = (gcnew System::Windows::Forms::ProgressBar());

this->trackBar2 = (gcnew System::Windows::Forms::TrackBar());

this->Punch1 = (gcnew System::Windows::Forms::Button());

this->Punch2 = (gcnew System::Windows::Forms::Button());

this->Roll1 = (gcnew System::Windows::Forms::Button());

this->Punch3 = (gcnew System::Windows::Forms::Button());

this->Punch4 = (gcnew System::Windows::Forms::Button());

this->Roll3 = (gcnew System::Windows::Forms::Button());

this->panel1 = (gcnew System::Windows::Forms::Panel());

this->label3 = (gcnew System::Windows::Forms::Label());

this->checkBox3 = (gcnew System::Windows::Forms::CheckBox());

this->checkBox2 = (gcnew System::Windows::Forms::CheckBox());

this->checkBox1 = (gcnew System::Windows::Forms::CheckBox());

this->panel5 = (gcnew System::Windows::Forms::Panel());

this->Roll2 = (gcnew System::Windows::Forms::Button());

this->Roll4 = (gcnew System::Windows::Forms::Button());

this->panel3New = (gcnew System::Windows::Forms::Panel());

this->panel3Ex = (gcnew System::Windows::Forms::Panel());

this->panel2New = (gcnew System::Windows::Forms::Panel());

this->panel2Ex = (gcnew System::Windows::Forms::Panel());

this->panel1New = (gcnew System::Windows::Forms::Panel());

this->panel1Ex = (gcnew System::Windows::Forms::Panel());

this->label1 = (gcnew System::Windows::Forms::Label());

this->timer1 = (gcnew System::Windows::Forms::Timer(this->components));

this->label2 = (gcnew System::Windows::Forms::Label());

this->NextLevel = (gcnew System::Windows::Forms::Button());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar1))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar2))->BeginInit();

this->panel1->SuspendLayout();

this->panel5->SuspendLayout();

this->SuspendLayout();

//

// trackBar1

//

this->trackBar1->Enabled = false;

this->trackBar1->Location = System::Drawing::Point(12, 266);

this->trackBar1->Margin = System::Windows::Forms::Padding(0);

this->trackBar1->Maximum = 1000;

this->trackBar1->Name = L"trackBar1";

this->trackBar1->Size = System::Drawing::Size(380, 45);

this->trackBar1->TabIndex = 1;

this->trackBar1->TickStyle = System::Windows::Forms::TickStyle::None;

//

// progressBar1

//

this->progressBar1->BackColor = System::Drawing::SystemColors::ActiveCaption;

this->progressBar1->Location = System::Drawing::Point(125, 336);

this->progressBar1->Name = L"progressBar1";

this->progressBar1->Size = System::Drawing::Size(266, 13);

this->progressBar1->TabIndex = 2;

this->progressBar1->Value = 100;

//

// trackBar2

//

this->trackBar2->BackColor = System::Drawing::SystemColors::ControlDark;

this->trackBar2->Enabled = false;

this->trackBar2->Location = System::Drawing::Point(12, 288);

this->trackBar2->Margin = System::Windows::Forms::Padding(0);

this->trackBar2->Maximum = 1000;

this->trackBar2->Name = L"trackBar2";

this->trackBar2->Size = System::Drawing::Size(380, 45);

this->trackBar2->TabIndex = 3;

this->trackBar2->TickStyle = System::Windows::Forms::TickStyle::TopLeft;

//

// Punch1

//

this->Punch1->BackColor = System::Drawing::Color::White;

this->Punch1->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"Punch1.BackgroundImage")));

this->Punch1->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->Punch1->Location = System::Drawing::Point(72, 13);

this->Punch1->Name = L"Punch1";

this->Punch1->Size = System::Drawing::Size(40, 40);

this->Punch1->TabIndex = 0;

this->Punch1->UseVisualStyleBackColor = false;

this->Punch1->Click += gcnew System::EventHandler(this, &GameMetalSmith::button1\_Click);

//

// Punch2

//

this->Punch2->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"Punch2.BackgroundImage")));

this->Punch2->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->Punch2->Location = System::Drawing::Point(118, 13);

this->Punch2->Name = L"Punch2";

this->Punch2->Size = System::Drawing::Size(40, 40);

this->Punch2->TabIndex = 0;

this->Punch2->UseVisualStyleBackColor = true;

this->Punch2->Click += gcnew System::EventHandler(this, &GameMetalSmith::button2\_Click);

//

// Roll1

//

this->Roll1->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"Roll1.BackgroundImage")));

this->Roll1->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->Roll1->Location = System::Drawing::Point(176, 12);

this->Roll1->Name = L"Roll1";

this->Roll1->Size = System::Drawing::Size(40, 40);

this->Roll1->TabIndex = 0;

this->Roll1->UseVisualStyleBackColor = true;

this->Roll1->Click += gcnew System::EventHandler(this, &GameMetalSmith::button3\_Click);

//

// Punch3

//

this->Punch3->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"Punch3.BackgroundImage")));

this->Punch3->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->Punch3->Location = System::Drawing::Point(72, 58);

this->Punch3->Name = L"Punch3";

this->Punch3->Size = System::Drawing::Size(40, 40);

this->Punch3->TabIndex = 0;

this->Punch3->UseVisualStyleBackColor = true;

this->Punch3->Click += gcnew System::EventHandler(this, &GameMetalSmith::button4\_Click);

//

// Punch4

//

this->Punch4->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"Punch4.BackgroundImage")));

this->Punch4->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->Punch4->Location = System::Drawing::Point(118, 58);

this->Punch4->Name = L"Punch4";

this->Punch4->Size = System::Drawing::Size(40, 40);

this->Punch4->TabIndex = 0;

this->Punch4->UseVisualStyleBackColor = true;

this->Punch4->Click += gcnew System::EventHandler(this, &GameMetalSmith::button5\_Click);

//

// Roll3

//

this->Roll3->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"Roll3.BackgroundImage")));

this->Roll3->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->Roll3->Location = System::Drawing::Point(176, 58);

this->Roll3->Name = L"Roll3";

this->Roll3->Size = System::Drawing::Size(40, 40);

this->Roll3->TabIndex = 0;

this->Roll3->UseVisualStyleBackColor = true;

this->Roll3->Click += gcnew System::EventHandler(this, &GameMetalSmith::button6\_Click);

//

// panel1

//

this->panel1->Controls->Add(this->label3);

this->panel1->Controls->Add(this->checkBox3);

this->panel1->Controls->Add(this->checkBox2);

this->panel1->Controls->Add(this->checkBox1);

this->panel1->Controls->Add(this->panel5);

this->panel1->Controls->Add(this->panel3New);

this->panel1->Controls->Add(this->panel3Ex);

this->panel1->Controls->Add(this->panel2New);

this->panel1->Controls->Add(this->panel2Ex);

this->panel1->Controls->Add(this->panel1New);

this->panel1->Controls->Add(this->panel1Ex);

this->panel1->Location = System::Drawing::Point(12, 12);

this->panel1->Name = L"panel1";

this->panel1->Size = System::Drawing::Size(379, 248);

this->panel1->TabIndex = 4;

//

// label3

//

this->label3->AutoSize = true;

this->label3->BackColor = System::Drawing::Color::DarkRed;

this->label3->Font = (gcnew System::Drawing::Font(L"Arial Black", 30, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label3->ForeColor = System::Drawing::Color::Gold;

this->label3->Location = System::Drawing::Point(15, 150);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(0, 56);

this->label3->TabIndex = 7;

//

// checkBox3

//

this->checkBox3->AutoSize = true;

this->checkBox3->Enabled = false;

this->checkBox3->Location = System::Drawing::Point(229, 56);

this->checkBox3->Name = L"checkBox3";

this->checkBox3->Size = System::Drawing::Size(15, 14);

this->checkBox3->TabIndex = 2;

this->checkBox3->UseVisualStyleBackColor = true;

//

// checkBox2

//

this->checkBox2->AutoSize = true;

this->checkBox2->Enabled = false;

this->checkBox2->Location = System::Drawing::Point(183, 56);

this->checkBox2->Name = L"checkBox2";

this->checkBox2->Size = System::Drawing::Size(15, 14);

this->checkBox2->TabIndex = 2;

this->checkBox2->UseVisualStyleBackColor = true;

//

// checkBox1

//

this->checkBox1->AutoSize = true;

this->checkBox1->Enabled = false;

this->checkBox1->Location = System::Drawing::Point(137, 56);

this->checkBox1->Name = L"checkBox1";

this->checkBox1->Size = System::Drawing::Size(15, 14);

this->checkBox1->TabIndex = 2;

this->checkBox1->UseVisualStyleBackColor = true;

this->checkBox1->CheckedChanged += gcnew System::EventHandler(this, &GameMetalSmith::checkBox1\_CheckedChanged);

//

// panel5

//

this->panel5->Controls->Add(this->Punch1);

this->panel5->Controls->Add(this->Punch4);

this->panel5->Controls->Add(this->Roll1);

this->panel5->Controls->Add(this->Roll2);

this->panel5->Controls->Add(this->Roll4);

this->panel5->Controls->Add(this->Punch2);

this->panel5->Controls->Add(this->Roll3);

this->panel5->Controls->Add(this->Punch3);

this->panel5->Location = System::Drawing::Point(20, 123);

this->panel5->Name = L"panel5";

this->panel5->Size = System::Drawing::Size(334, 111);

this->panel5->TabIndex = 1;

//

// Roll2

//

this->Roll2->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"Roll2.BackgroundImage")));

this->Roll2->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->Roll2->Location = System::Drawing::Point(222, 12);

this->Roll2->Name = L"Roll2";

this->Roll2->Size = System::Drawing::Size(40, 40);

this->Roll2->TabIndex = 0;

this->Roll2->UseVisualStyleBackColor = true;

this->Roll2->Click += gcnew System::EventHandler(this, &GameMetalSmith::button7\_Click);

//

// Roll4

//

this->Roll4->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"Roll4.BackgroundImage")));

this->Roll4->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->Roll4->Location = System::Drawing::Point(222, 58);

this->Roll4->Name = L"Roll4";

this->Roll4->Size = System::Drawing::Size(40, 40);

this->Roll4->TabIndex = 0;

this->Roll4->UseVisualStyleBackColor = true;

this->Roll4->Click += gcnew System::EventHandler(this, &GameMetalSmith::button8\_Click);

//

// panel3New

//

this->panel3New->BackColor = System::Drawing::Color::SlateGray;

this->panel3New->Location = System::Drawing::Point(215, 76);

this->panel3New->Name = L"panel3New";

this->panel3New->Size = System::Drawing::Size(40, 40);

this->panel3New->TabIndex = 1;

//

// panel3Ex

//

this->panel3Ex->BackColor = System::Drawing::Color::SlateGray;

this->panel3Ex->Location = System::Drawing::Point(215, 10);

this->panel3Ex->Name = L"panel3Ex";

this->panel3Ex->Size = System::Drawing::Size(40, 40);

this->panel3Ex->TabIndex = 1;

//

// panel2New

//

this->panel2New->BackColor = System::Drawing::Color::SlateGray;

this->panel2New->Location = System::Drawing::Point(169, 76);

this->panel2New->Name = L"panel2New";

this->panel2New->Size = System::Drawing::Size(40, 40);

this->panel2New->TabIndex = 1;

//

// panel2Ex

//

this->panel2Ex->BackColor = System::Drawing::Color::SlateGray;

this->panel2Ex->Location = System::Drawing::Point(169, 10);

this->panel2Ex->Name = L"panel2Ex";

this->panel2Ex->Size = System::Drawing::Size(40, 40);

this->panel2Ex->TabIndex = 1;

//

// panel1New

//

this->panel1New->BackColor = System::Drawing::Color::SlateGray;

this->panel1New->Location = System::Drawing::Point(123, 76);

this->panel1New->Name = L"panel1New";

this->panel1New->Size = System::Drawing::Size(40, 40);

this->panel1New->TabIndex = 1;

//

// panel1Ex

//

this->panel1Ex->BackColor = System::Drawing::Color::SlateGray;

this->panel1Ex->Location = System::Drawing::Point(123, 10);

this->panel1Ex->Name = L"panel1Ex";

this->panel1Ex->Size = System::Drawing::Size(40, 40);

this->panel1Ex->TabIndex = 1;

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(9, 407);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(0, 13);

this->label1->TabIndex = 5;

//

// timer1

//

this->timer1->Interval = 1000;

this->timer1->Tick += gcnew System::EventHandler(this, &GameMetalSmith::timer1\_Tick);

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(14, 336);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(105, 13);

this->label2->TabIndex = 5;

this->label2->Text = L"Времени осталось:";

//

// NextLevel

//

this->NextLevel->Location = System::Drawing::Point(262, 355);

this->NextLevel->Name = L"NextLevel";

this->NextLevel->Size = System::Drawing::Size(129, 23);

this->NextLevel->TabIndex = 6;

this->NextLevel->Text = L"Следующий уровень";

this->NextLevel->UseVisualStyleBackColor = true;

this->NextLevel->Click += gcnew System::EventHandler(this, &GameMetalSmith::button9\_Click);

//

// GameMetalSmith

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::SystemColors::ControlDark;

this->ClientSize = System::Drawing::Size(406, 387);

this->Controls->Add(this->NextLevel);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->panel1);

this->Controls->Add(this->trackBar2);

this->Controls->Add(this->trackBar1);

this->Controls->Add(this->progressBar1);

this->Name = L"GameMetalSmith";

this->Text = L"Blacksmith";

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar1))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar2))->EndInit();

this->panel1->ResumeLayout(false);

this->panel1->PerformLayout();

this->panel5->ResumeLayout(false);

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void timer1\_Tick(System::Object^ sender, System::EventArgs^ e) {

if(progressBar1->Value - 1 >= 0) progressBar1->Value -= 1;

else {

label3->Text = "Вы проиграли!";

timer1->Enabled = false;

NextLevel->Enabled = true;

round = 1;

label1->Text = "";

(gcnew System::Media::SoundPlayer("wavs\\zhil\_do\_konsta.wav"))->Play();;

}

}

private: System::Void button9\_Click(System::Object^ sender, System::EventArgs^ e) {

checkBox1->Checked = false;

checkBox2->Checked = false;

checkBox3->Checked = false;

NextLevel->Enabled = false;

progressBar1->Value = progressBar1->Maximum;

timer1->Enabled = true;

label1->Text = "Раунд: " + (round++).ToString();

label3->Text = "";

srand(time(0));

rand();

int amount = rand() % 21 + 10, tempSum = 50, tempIndex;;

Int32 tempPunch;

for (int i = 0; i < amount; i++) {

do {

tempPunch = punches[tempIndex = rand() % 8];

} while (((tempSum + tempPunch) > 950) || ((tempSum + tempPunch) < 50));

tempSum += tempPunch;

switch (amount - i)

{

case 1:

last[0] = tempIndex;

panel1Ex->BackgroundImage = Image::FromFile(punchesImages[tempIndex]);

break;

case 2:

last[1] = tempIndex;

panel2Ex->BackgroundImage = Image::FromFile(punchesImages[tempIndex]);

break;

case 3:

last[2] = tempIndex;

panel3Ex->BackgroundImage = Image::FromFile(punchesImages[tempIndex]);

break;

default: break;

}

}

trackBar1->Value = tempSum;

timer1->Enabled = true;

}

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) {

int i = 0;

(gcnew System::Media::SoundPlayer("wavs\\1p.wav"))->Play();

if ((trackBar2->Value + punches[i] <= trackBar2->Maximum) && (trackBar2->Value + punches[i] >= trackBar2->Minimum)) {

trackBar2->Value += punches[i];

panel3New->BackgroundImage = panel2New->BackgroundImage;

panel2New->BackgroundImage = panel1New->BackgroundImage;

panel1New->BackgroundImage = Image::FromFile(punchesImages[i]);

switch (streak)

{

case 0:

if (last[2] == i) {

streak++;

checkBox3->Checked = true;

checkBox2->Checked = false;

checkBox1->Checked = false;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 1:

if (last[1] == i) {

streak++;

checkBox2->Checked = true;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\ay\_ay\_ay.wav"))->Play();

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 2:

if (last[0] == i) {

streak++;

checkBox1->Checked = true;

}

else {

(gcnew System::Media::SoundPlayer("wavs\\bozhe\_kak\_eto\_bolno.wav"))->Play();

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

}

}

}

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e) {

int i = 1;

(gcnew System::Media::SoundPlayer("wavs\\1p.wav"))->Play();

if ((trackBar2->Value + punches[i] <= trackBar2->Maximum) && (trackBar2->Value + punches[i] >= trackBar2->Minimum)) {

trackBar2->Value += punches[i];

panel3New->BackgroundImage = panel2New->BackgroundImage;

panel2New->BackgroundImage = panel1New->BackgroundImage;

panel1New->BackgroundImage = Image::FromFile(punchesImages[i]);

switch (streak)

{

case 0:

if (last[2] == i) {

streak++;

checkBox3->Checked = true;

checkBox2->Checked = false;

checkBox1->Checked = false;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 1:

if (last[1] == i) {

streak++;

checkBox2->Checked = true;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\ay\_ay\_ay.wav"))->Play();

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 2:

if (last[0] == i) {

streak++;

checkBox1->Checked = true;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\bozhe\_kak\_eto\_bolno.wav"))->Play();

streak = 0;

}

break;

}

}

}

private: System::Void button4\_Click(System::Object^ sender, System::EventArgs^ e) {

int i = 2;

(gcnew System::Media::SoundPlayer("wavs\\1p.wav"))->Play();

if ((trackBar2->Value + punches[i] <= trackBar2->Maximum) && (trackBar2->Value + punches[i] >= trackBar2->Minimum)) {

trackBar2->Value += punches[i];

panel3New->BackgroundImage = panel2New->BackgroundImage;

panel2New->BackgroundImage = panel1New->BackgroundImage;

panel1New->BackgroundImage = Image::FromFile(punchesImages[i]);

switch (streak)

{

case 0:

if (last[2] == i) {

streak++;

checkBox3->Checked = true;

checkBox2->Checked = false;

checkBox1->Checked = false;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 1:

if (last[1] == i) {

streak++;

checkBox2->Checked = true;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\ay\_ay\_ay.wav"))->Play();

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 2:

if (last[0] == i) {

streak++;

checkBox1->Checked = true;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\bozhe\_kak\_eto\_bolno.wav"))->Play();

streak = 0;

}

break;

}

}

}

private: System::Void button5\_Click(System::Object^ sender, System::EventArgs^ e) {

int i = 3;

(gcnew System::Media::SoundPlayer("wavs\\1p.wav"))->Play();

if ((trackBar2->Value + punches[i] <= trackBar2->Maximum) && (trackBar2->Value + punches[i] >= trackBar2->Minimum)) {

trackBar2->Value += punches[i];

panel3New->BackgroundImage = panel2New->BackgroundImage;

panel2New->BackgroundImage = panel1New->BackgroundImage;

panel1New->BackgroundImage = Image::FromFile(punchesImages[i]);

switch (streak)

{

case 0:

if (last[2] == i) {

streak++;

checkBox3->Checked = true;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 1:

if (last[1] == i) {

streak++;

checkBox2->Checked = true;

checkBox2->Checked = false;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\ay\_ay\_ay.wav"))->Play();

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 2:

if (last[0] == i) {

streak++;

checkBox1->Checked = true;

checkBox1->Checked = false;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

(gcnew System::Media::SoundPlayer("wavs\\bozhe\_kak\_eto\_bolno.wav"))->Play();

}

break;

}

}

}

private: System::Void button3\_Click(System::Object^ sender, System::EventArgs^ e) {

int i = 4;

(gcnew System::Media::SoundPlayer("wavs\\1p.wav"))->Play();

if ((trackBar2->Value + punches[i] <= trackBar2->Maximum) && (trackBar2->Value + punches[i] >= trackBar2->Minimum)) {

trackBar2->Value += punches[i];

panel3New->BackgroundImage = panel2New->BackgroundImage;

panel2New->BackgroundImage = panel1New->BackgroundImage;

panel1New->BackgroundImage = Image::FromFile(punchesImages[i]);

switch (streak)

{

case 0:

if (last[2] == i) {

streak++;

checkBox3->Checked = true;

checkBox2->Checked = false;

checkBox1->Checked = false;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 1:

if (last[1] == i) {

streak++;

checkBox2->Checked = true;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\ay\_ay\_ay.wav"))->Play();

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 2:

if (last[0] == i) {

streak++;

checkBox1->Checked = true;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

(gcnew System::Media::SoundPlayer("wavs\\bozhe\_kak\_eto\_bolno.wav"))->Play();

}

break;

}

}

}

private: System::Void button7\_Click(System::Object^ sender, System::EventArgs^ e) {

int i = 5;

(gcnew System::Media::SoundPlayer("wavs\\1p.wav"))->Play();

if ((trackBar2->Value + punches[i] <= trackBar2->Maximum) && (trackBar2->Value + punches[i] >= trackBar2->Minimum)) {

trackBar2->Value += punches[i];

panel3New->BackgroundImage = panel2New->BackgroundImage;

panel2New->BackgroundImage = panel1New->BackgroundImage;

panel1New->BackgroundImage = Image::FromFile(punchesImages[i]);

switch (streak)

{

case 0:

if (last[2] == i) {

streak++;

checkBox3->Checked = true;

checkBox2->Checked = false;

checkBox1->Checked = false;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 1:

if (last[1] == i) {

streak++;

checkBox2->Checked = true;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\ay\_ay\_ay.wav"))->Play();

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 2:

if (last[0] == i) {

streak++;

checkBox1->Checked = true;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

(gcnew System::Media::SoundPlayer("wavs\\bozhe\_kak\_eto\_bolno.wav"))->Play();

}

break;

}

}

}

private: System::Void button6\_Click(System::Object^ sender, System::EventArgs^ e) {

int i = 6;

(gcnew System::Media::SoundPlayer("wavs\\1p.wav"))->Play();

if ((trackBar2->Value + punches[i] <= trackBar2->Maximum) && (trackBar2->Value + punches[i] >= trackBar2->Minimum)) {

trackBar2->Value += punches[i];

panel3New->BackgroundImage = panel2New->BackgroundImage;

panel2New->BackgroundImage = panel1New->BackgroundImage;

panel1New->BackgroundImage = Image::FromFile(punchesImages[i]);

switch (streak)

{

case 0:

if (last[2] == i) {

streak++;

checkBox3->Checked = true;

checkBox2->Checked = false;

checkBox1->Checked = false;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 1:

if (last[1] == i) {

streak++;

checkBox2->Checked = true;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\ay\_ay\_ay.wav"))->Play();

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 2:

if (last[0] == i) {

streak++;

checkBox1->Checked = true;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

(gcnew System::Media::SoundPlayer("wavs\\bozhe\_kak\_eto\_bolno.wav"))->Play();

}

break;

}

}

}

private: System::Void button8\_Click(System::Object^ sender, System::EventArgs^ e) {

int i = 7;

(gcnew System::Media::SoundPlayer("wavs\\1p.wav"))->Play();

if ((trackBar2->Value + punches[i] <= trackBar2->Maximum) && (trackBar2->Value + punches[i] >= trackBar2->Minimum)) {

trackBar2->Value += punches[i];

panel3New->BackgroundImage = panel2New->BackgroundImage;

panel2New->BackgroundImage = panel1New->BackgroundImage;

panel1New->BackgroundImage = Image::FromFile(punchesImages[i]);

switch (streak)

{

case 0:

if (last[2] == i) {

streak++;

checkBox3->Checked = true;

checkBox2->Checked = false;

checkBox1->Checked = false;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 1:

if (last[1] == i) {

streak++;

checkBox2->Checked = true;

checkBox1->Checked = false;

(gcnew System::Media::SoundPlayer("wavs\\ay\_ay\_ay.wav"))->Play();

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

}

break;

case 2:

if (last[0] == i) {

streak++;

checkBox1->Checked = true;

}

else {

checkBox3->Checked = false;

checkBox2->Checked = false;

checkBox1->Checked = false;

streak = 0;

(gcnew System::Media::SoundPlayer("wavs\\bozhe\_kak\_eto\_bolno.wav"))->Play();

}

break;

}

}

}

private: System::Void checkBox1\_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {

if (trackBar1->Value == trackBar2->Value) {

label3->Text = "Вы выйграли";

timer1->Enabled = false;

NextLevel->Enabled = true;

progressBar1->Value = progressBar1->Maximum;

(gcnew System::Media::SoundPlayer("wavs\\krasavchik.wav"))->Play();

}

else {

streak = 0;

(gcnew System::Media::SoundPlayer("wavs\\eto\_nenormalno.wav"))->Play();

}

}

};

}

About.h

#pragma once

namespace Base {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для About

/// </summary>

public ref class About : public System::Windows::Forms::Form

{

public:

About(void)

{

InitializeComponent();

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~About()

{

if (components)

{

delete components;

}

}

protected:

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::Label^ label2;

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(About::typeid));

this->label1 = (gcnew System::Windows::Forms::Label());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->label2 = (gcnew System::Windows::Forms::Label());

this->SuspendLayout();

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 15, static\_cast<System::Drawing::FontStyle>((System::Drawing::FontStyle::Bold | System::Drawing::FontStyle::Underline)),

System::Drawing::GraphicsUnit::Point, static\_cast<System::Byte>(204)));

this->label1->Location = System::Drawing::Point(6, 6);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(150, 25);

this->label1->TabIndex = 0;

this->label1->Text = L"О программе";

//

// richTextBox1

//

this->richTextBox1->Location = System::Drawing::Point(12, 40);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->ReadOnly = true;

this->richTextBox1->ScrollBars = System::Windows::Forms::RichTextBoxScrollBars::Vertical;

this->richTextBox1->Size = System::Drawing::Size(398, 304);

this->richTextBox1->TabIndex = 1;

this->richTextBox1->Text = resources->GetString(L"richTextBox1.Text");

//

// label2

//

this->label2->AutoSize = true;

this->label2->Cursor = System::Windows::Forms::Cursors::Default;

this->label2->ForeColor = System::Drawing::SystemColors::HotTrack;

this->label2->Location = System::Drawing::Point(8, 347);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(183, 13);

this->label2->TabIndex = 2;

this->label2->Text = L"Разработчик: Prikhodko Pavel 2018";

// About

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(419, 374);

this->Controls->Add(this->label2);

this->Controls->Add(this->richTextBox1);

this->Controls->Add(this->label1);

this->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::FixedDialog;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"About";

this->Text = L"Справка";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

};

}

# **Приложение В Руководство системного администратора**

На переносном диске содержится архивированная папка с программой и ее исходным кодом. Для работы с программой, необходимо разархивировать данную папку, найти и запустить файл Base.exe. Программа готова к использованию.

Для функционирования программы требуется:

* ОС Windows XP SP2 или выше;
* Оперативаная память: 15МБ и выше;
* Жесткий диск: 3.5МБ для исполняемого файла и вспомогательных файлов; допольнительное место для сохранения файлов;
* .NET Framework версии 2.0 или выше;
* Наличие мыши, клавиатуры, монитора, устройства вывода звука(необязательно);
* Разрешение экрана 800 x 600 или более.

# **Приложение Г Руководство пользователя**

Для запуска программы необходимо 2 раза кликнуть по файлу Base.exe. В открывшемся окне будет предложено ввести логин и пароль, а так же создать новый аккаунт. При первом входе необходимо создать учетную запись. Для этого нужно сделать 1 клик по кнопке “Создать новую учетную запись”.

В открывшемся окне ввести логин, пароль и выбрать тип создаваемого аккаунта. Для получения справки следует навести курсор на надпись “Наведите, чтобы узнать больше”. После чего нажмите кнопку “Создать новую учетную запись”.

Для начала работы войдите в созданную учетную запись (введите логин и пароль в соответствующие поля и нажмите кнопку войти). В открывшемся окне, для будут видны Меню и Таблица, в которой будут отображаться данные из файла. Для любого типа аккаунта доступны следующие функции:

* Открытие файла: (Файл -> Открыть) открывается уже созданный файл, данные из которого будут отображены в таблице после открытия
* Выполнение запроса: (Запрос -> Создать запрос) в открывшемся окне, необходимо отметить поля необходимые для выборки поля, после чего ввести в эти поля необходимую информацию. Для выполнения запроса следует нажать кнопку “Выполнить запрос”. В таблице будут отображены результаты запроса. Для обновления таблицы следует нажать кнопку “Сбросить запрос”
* Игра – “Кузнец”. Суть игры: совместить верхний и нижний ползунок. После нажатия на кнопку “Следующий уровень” начнется отсчет, за время которого следует, используя крысные и зеленые кнопки двигать нижний ползунок, чтобы он совместился с верхним, причем последними должны быть нанесены те удары, которые отображены на верхних 3 картинках. Игра является бесконечной. Игра заканчивается, по окончании времени, после чего можно начать новую игру, нажав кнопку “Следующий уровень”. Игра имеет звуковое сопровождение.
* О программе – (О программе -> О программе) В открывшемся окне предоставляется общая информация о программе
* Смена учетной записи: (Файл -> Сменить учетную запись) в открывшемся окне появятся поля для ввода логина и пароля. Для возврата к файлу, необходимо нажать на крестик в правом верхнем углу окна

Также для аккаунтов типа “Администратор” добавляются специальный функционал:

* Создание нового документа: (Файл -> Создать) В открывшемся окне указывается директория и имя файла, который будет создан
* Сохранение изменений: (Файл -> Сохранить / Сохранить как) при выборе “Сохраниь” сохраняются все изменения в текущем файле. При выборе “Сохранить как” в открывшемся окне можно указать тот файл, который будет перезаписан, либо создать новый файл, в который будут сохранены данные из текущего.
* Редактирование
  + Добавить новую запись: (Редактировать -> Добавить новую запись) в открывшемся окне появятся поля ввода данных для новой записи. Для того чтобы добавить запись в таблицу, следует нажать кнопку “Добавить”
  + Редактировать существующую запись: (Редактировать -> Редактировать) для выбора редактируемой записи следует выделить ее в главном окне, путем нажатия на нее левой кнопкой мыши, после чего выбрать пункт Редкатировать. В открывшемся окне в полях будут введены текущие занчения полей записи. После редактирования полей, для сохранения внесенных изменений необходимо нажать “Изменить”
  + Удалить запись (Редактировать -> Удалить) для выбора удаляемой записи следует выделить ее в главном окне, путем нажатия на нее левой кнопкой мыши, после чего выбрать пункт Удалить. В открывшемся окне появится предупреждение об удалении. После согласия запись будет безвозвратно удалена.