Министерство науки и высшего образования 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\_\_\_\_\_\_\_\_\_

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019г.

Руководитель от вуза

**\_\_\_\_\_\_\_\_\_\_\_**\_/Егорова Е.В./

*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-81\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Бачище О.И.

*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.*

2019

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 ПИ81

(Ф.И.О.)

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

(наименование)

Сроки практики 17.06.2019 г. по 30.06.2019 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** |
| 11 | 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 |
| 22 | Знакомство с MSDN. Самостоятельная работа с документацией | 1 неделя | Формирование компетенции ОК-7: способность к самоорганизации и самообразованию |
| 33 | Разработка приложения с графическим интерфейсом по индивидуальному заданию | 2 неделя | Формирование компетенции ПК-1: готовность применять основные методы и инструменты разработки программного обеспечения |
| 44 | 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., доцент  (подпись) (Ф.И.О., должность) |  |
| Руководитель практики от  профильной организации \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Егорова Е.В., доцент  (подпись) (Ф.И.О., должность) |  |

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

(подпись) (Ф.И.О.)

**Инструктаж по ОТ, ТБ, ПБ, ПВТР**

Инструктаж обучающегося по ознакомлению с требованиями охраны труда, техники безопасности, пожарной безопасности, а также правилами внутреннего трудового распорядка проведен «\_\_\_» \_\_\_\_\_\_\_\_\_ 2019 г.

Руководитель практики от

профильной организации \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_Егорова Е.В., доцент\_\_

(подпись) (Ф.И.О., должность)

МП

Содержание

[Введение 4](#_Toc11631756)

[1. Техническое задание 5](#_Toc11631757)

[1.1 Терминология 5](#_Toc11631758)

[1.2 Требования к функциональности 6](#_Toc11631759)

[2 Проект программного продукта 7](#_Toc11631760)

[2.1 Структура данных 7](#_Toc11631761)

[2.2 Структура файлов 7](#_Toc11631762)

[3 Описание программного продукта 8](#_Toc11631763)

[3.1 Общий подход к программной реализации 8](#_Toc11631764)

[3.2 Модульная структура программного обеспечения 8](#_Toc11631765)

[3.3 Описание реализации ПО 10](#_Toc11631766)

[Заключение 12](#_Toc11631767)

[Список использованной литературы 13](#_Toc11631768)

[Приложение А Снимки экранных форм пользовательского интерфейса 14](#_Toc11631769)

[Приложение Б Исходный код программы 17](#_Toc11631770)

[Приложение В Руководство системного администратора 50](#_Toc11631771)

[Приложение Г Руководство пользователя 52](#_Toc11631772)

# Введение

Для коммерческих проектов всегда была актуальна проблема хранения информации, её структуризация и возможность эффективного использования. Сегодня эта задача может быть значительно упрощена за счёт работы с электронными таблицами и базами данных.

Работа с пользовательскими данными – это большая ответственность. Поэтому каждому разработчику необходимо получить опыт в реализации подобных проектов, понять тонкости использования заданного редактора кода, изучить основную документацию и научиться эффективно решать поставленные перед собой задачи.

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

**Главная задача**: написание программного продукта с удобным интерфейсом, позволяющего администратору в защищенном режиме редактировать информацию базы данных Аэропорта, а пользователю эффективно взаимодействовать с ней.

# Техническое задание

Организовать работу с базой данных, состоящих из строк. Каждая строка содержит:

* + Номер рейса;
  + Тип самолёта;
  + Пункт назначения;
  + Дни отправления;
  + Время вылета;
  + Время прилёта;
  + Цена билета.

Например: 121, Ил96, Москва, 1,2,5 10.00 14.00 3000.20.

Реализовать обработку файла с расписанием самолётов (добавление, удаление, редактирование записей, выполнение запросов по поиску), используя среду разработки Visual Studio и визуальные компоненты Windows Forms C++. Дополнительно: разработка игры.

## Терминология

Запись – строка, содержащая информацию об одной игрушке.

Запись содержит следующие поля:

* + Номер рейса – целочисленное число;
  + Тип самолёта – строка, состоящая из букв и цифр;
  + Пункт назначения – наименование города – строка, состоящая из букв;
  + Дни отправления – строа из нескольких целочисленных элементов, разделённых запятыми;
  + Время вылета – время в формате 23:59;
  + Время прилёта – время в формате 23:59;
  + Цена билета – число в формате с плавающей точкой.

*Таблица* содержит множество строк – записей.

Функция *редактирование* служит для изменения отдельных записей таблицы.

*Удаление* служит для исключения записей из таблицы

*Запрос* служит для поиска определенных записей по условиям, которые задал пользователь.

*Добавление* служит для внесения новых записей в таблицу

*БД – база данных* – совокупность всех записей таблицы

## Требования к функциональности

Главная форма должна быть окном авторизации и содержать кнопки с выбором прав доступа для работы с таблицей: администратор (имеет расширенный функционал работы с БД, для подтверждения необходимо ввести пароль) и пользователь

Форма, взаимодействующая с базой данных, должна содержать таблицу с записями и меню с элементами, имеющими следющие пункты для всех прав доступа:

1. Справка;
2. Запрос;
3. Игра;
4. Выход из БД;

А также расширенный функционал, доступный только администратору:

1. Редактирование
   1. Редакировать запись;
   2. Добавить запись;
   3. Удалить запись;
2. Работа с БД
   1. Сохранить изменения;
   2. Изменить пароль;

Необходимые функции работы с записями:

* Добавление новой записи – форма, в которой необходимо заполнить все поля и нажать на кнопку добавить.
* Удаление записи – форма, позволяющая удалить выделенные в столбце в формате CheckBox строки
* Редактирование записи – форма, вызывающая окно редактирования для каждой строки, выделенной в столбце формата CheckBox. Окно редактирования уже содержит внесенные в строку данные и позволяет форматированно редактировать любые из них
* Поля для запроса скрыты в форме, но, по при нажатии вызывающего их пункта меню, они становятся видимы пользователю и позволяют ему в реальном режиме увидеть результат на универсальный запрос.

# 2 Проект программного продукта

## 2.1 Структура данных

В данном продукте были реализованы:

Формы:

* login – главная форма проекта, содержит кнопки (объект Button) для выбора уровня доступа (Администратор или пользователь), а также кнопку выхода из программы и для вызова справки.
* main\_base – форма, содержащая таблицу с записями (отображается с помощью объекта dataGridView), горизонтальное меню (menuStrip), содержащее пункты для вызова справки, запроса, редактирования таблицы (с подпунктами для удаления, добавления и редактирования записей), работы с БД (сохранение файла и изменения пароля), вызова игры и выхода из БД.
* info – форма с основной информацией о программе
* make\_new\_pass – форма для создания нового пароля, содержащая о
* game – форма с игрой, управляющей самолётом “To Dublin”
* password – форма для ввода пароля, содержит TextBox для пользователя и кнопку для входа в БД
* plus\_string – форма для удаления и редактирования записей. Содержит maskedTextBox с маской для каждого поля, календарь с отображением текущего месяца для выбора дней отправления и кнопки для добавления/редактирования.

## 2.2 Структура файлов

Для работы программы необходим исполняемый файл программы.

Зашифрованный пароль хранится в файле password.txt, при его отсутствии пользователю предлагается задать новый пароль, который записывается в созданный автоматически файл. Записи базы данных хранятся построчно в файле base.txtБ при его отсутсвии он создается автоматически. Записи с рекордами игры хранятся в файле score.txt (при отсутсвии создаётся новый с таким же именем).

# 3 Описание программного продукта

## 3.1 Общий подход к программной реализации

Данное ПО реализовано на языке С++ с использованием среды разработки программного Microsoft Visual Studio 2019. Язык С++ - компилируемый статически типизированный язык программирования общего назначения.

Программа соответствует всем парадигмам объектно-ориентированного программирования. Для проверки введённых данных использовался элемент maskedTextBox, обработчики событий изменения текстового поля и ячеек таблицы в элемемнте DataGridView, элемент для выбора дат MonthCalendar, а также обработчики событий нажатия кнопок (button). Для шифровки и дешифровки прав досутпа к учетной записи администратора использовалось шифрование сдвигом Цезаря на заданное число.

## 3.2 Модульная структура программного обеспечения

В программе реализованы следующие модули и взаимодействие между ними, изображенные на рисунке 1 и на диаграмме UML (рисунок 2).

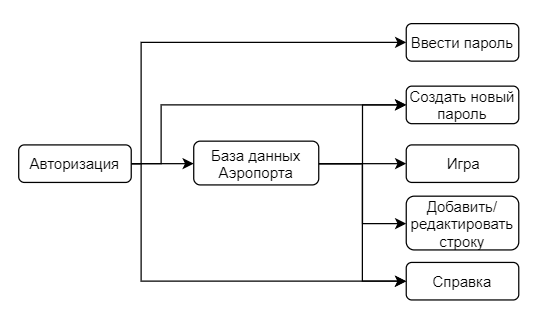


Рисунок 1 – Модульная структура

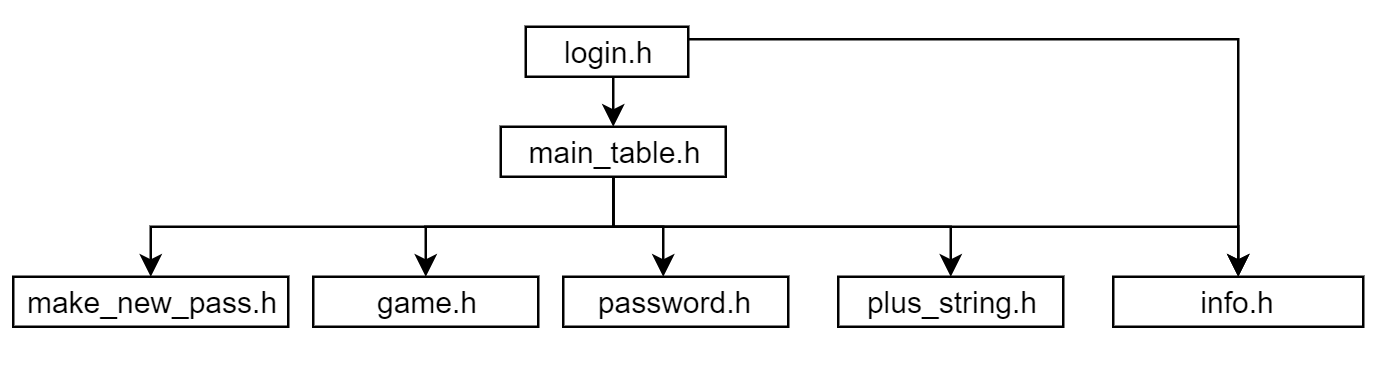


Рисунок 2 – Диаграмма классов

## 3.3 Описание реализации ПО

Главная форма (login.h) является окном авторизации и содержит кнопки с выбором прав доступа для работы с таблицей (button): администратор (имеет расширенный функционал работы с БД, для подтверждения необходимо ввести пароль) и пользователь. Для фиксирования нажатия кнопок используется обработчик событий Click.

После выбора прав доступа главная форма скрывается методом Hide() и вызывается форма для работы с таблицей “main\_base.h” (здесь и далее методом Show()). Если была нажата кнопка «Пользователь», то пункты меню, используемые для изменения данных таблицы (такие как «Добавить», «Изменить», «Сохранить», «Удалить» и т.д.), скрываются путем присваивания их свойству Visible значение false. Так же, для функционирования некоторых методов формы, зависящих от режима администрирования, необходим флаг is\_admin, который поднимается или опускается при её инициализации.

Форма, работающая с таблицей (“main\_base.h”) содержит в центре таблицу всех записей БД (реализовано с помощью dataGridView, доступно только для чтения; все значения изменяются программно), которая заполняется при загрузке формы из файла base.txt, а при её редактировании, путём считывания данных из объектов формы “plus\_string.h”. Сверху находится горизонтальное меню (реализовано с помощью menuStrip), состоящее из пунктов доступных всем пользователям: «Справка»; «Запрос»; «Игра»; «Выход из БД». А также расширенный функционал, доступный только администратору: «Редактирование» (с подпунктами «Редакировать запись»; «Добавить запись»; «Удалить запись); «Работа с БД» (с подпунктами «Сохранить изменения» и «Изменить пароль»).

При выборе пункта «Запрос», происходит увеличение видимой части формы (программно увеличивается значение Width). Также становятся видимы окна для ключевых слов запроса, по одному на каждый пункт строки.

Для пункта «Дни отправления» таким окном является элемент monthCalendar, предствляющий из себя календарь. Программно он транслирует только расписание текущего месяца. Выделенные для запроса даты отображаются в календаре жирным шрифтом (значение Bold). Значения выделенных дат (DateTime.Day) заносятся в целочисленный массив, который затем сравнивается со значениями ячеек в таблице, полученные после их конвертирования в формат System String и разбиения на лексемы методом

Split ().

Все остальные запросы заносятся в MaskedTextBox. Каждому из полей присвоена маска в соответсвии с техническим заданием.

Запрос реализуется методом Check(), который вызывается при изменении каждого из полей запроса (событие textChanged) и скрывает поля, не удовлетворяющие хотя бы одному из пунтов.

Форма добавления записи (“plus\_string.h”) состоит из тех же элементов, что и поля для запроса, но не позволяет создать форму если хотя бы одно из поллей заполнено неверно, и, в случае попытки сохранения некорректой строки, выдает сообщение об ошибке (MessageBox::Show()). Оно подсказывает, какую строку необходимо исправить.

В случае, если в поле для времени введено число большее 23:59, его значения заменяются на максимально возможные (например, запись 24:70 преобразуется в 23:59).

При вызове событий «Редактировать» и «Удалить» справа в таблице появляется дополнительный столбец в формате CheckBox, а снизу – кнопка для подтверждения действия. Выделенные строки подлежат удалению или редактированию в зависимости от метода, из вызвавщего. Для редактирования для каждой строки открывается форма “plus\_string”, в которой изменены поля с «Добавить» на «Редактировать», и в строки для данных уже внесены данные, содержащиеся в строке (после изменения они, уже отредактированные, будут внесены туда снова).

Крайний справа в режиме администрирования пункт меню представляет из себя иконку с изображением дискеты. В случае, если данные были изменены, но не сохранены, дискета меняет свой цвет на красный (программно, есть два пункта меню: с красной и черной иконкой, поочерёдно скрывающиеся/появляющиеся в зависимости от того, сохранён ли файл (флаг is\_save)). При наведении на черную иконку всплывает подсказка, что файл измненён, красная – подсказывает, что файл не сохранён и сохраняет при нажатии на него.

Форма с игрой (“game.h”) при запуске показывает кнопки “Играть”, “Правила” и “Рекорды”. При выборе любой из них кнопки пропадают и выполняется их соотвествующий функционал.

Пункт «Правила игры» скрывает все и показывает TextBox с правилами игры, возвращая всё на свои места по нажатию кнопки «Назад».

«Играть» запускается игра, представляющая из себя самолёт и 9 циклично отрисовывающихся по оси ОУ, но рандомно по оси ОХ облаков. Цель – не задеть самолёт. Скорость движения облаков увеличивается со временем. Рекорды также зависят от вермени. После окончания игры своё имя и набранный счёт предлагется занести в список с рекордами, просмотреть который можно в пункте «Рекорды».

# Заключение

Был создан программный продукт, позволяющий эффективно работать с базой данных Аэропорта после изучения краткой по объему документации. Реализованы защиты на ввод некорректных данных, эффективная обработка запроса в реальном режиме, без использования излишних форм.

В процессе выполнения задания были подробно изучены методы работы с файлами на языке С++, визуальными компонентами CLI, документацией MSDN.

Преимущества приложения:

* Интуитивно понятный интерфейс;
* Быстрая и эффективная обработка запроса;
* Возможность оперативного редактирования/ добавления/ удаления записей;
* Различные интерфейсы для пользователя и администратора;
* Приятный минималистичный дизайн;
* Увеличение сложности игры по мере прохождения;
* Шифрование пароля;
* Всплывающие и графические подсказки при работе с приложением.

Недочёты приложения:

* Работа только с одной таблицей;
* Отсутствие возможности выбора файла для обработки;
* Для работы необходимы специально отформатированные файлы.

Возможны дальнейшие усовершенствования программного продукта с течением времени:

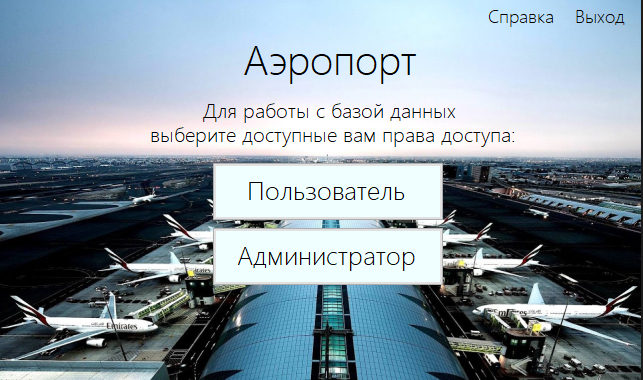
* Работа с несколькими таблицами одновременно;
* Расширение геймплея игры;
* Сохранение результатов запроса в файлы;
* Добавление функции печати файлов.

# Список использованной литературы

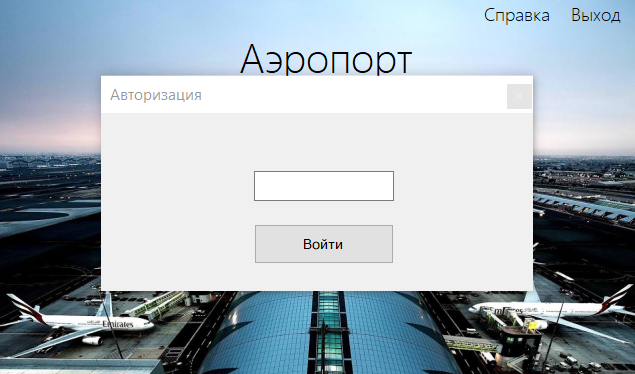
1. Пахомов Б.И. C/C++ и MS Visual C++ 2010 для начинающих. – СПб.: БХВ-Петербург, 2011. – 736с.
2. Егорова Е.В. Программирование на языке СИ. Учебное пособие/Алт. Госуд. Технич. Ун-т им. И.И.Ползунова.-Барнаул:2013.-184с.
3. “MSDN” – информационный сервис для разработчиков [Электронный ресурс] – Режим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

# Приложение А Снимки экранных форм пользовательского интерфейса

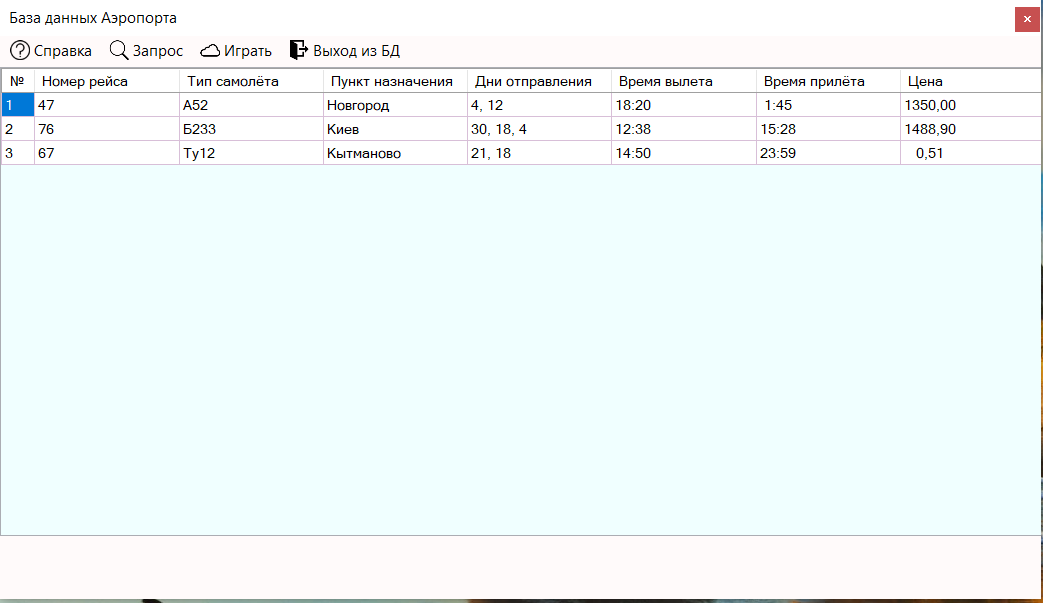
1. Главная форма



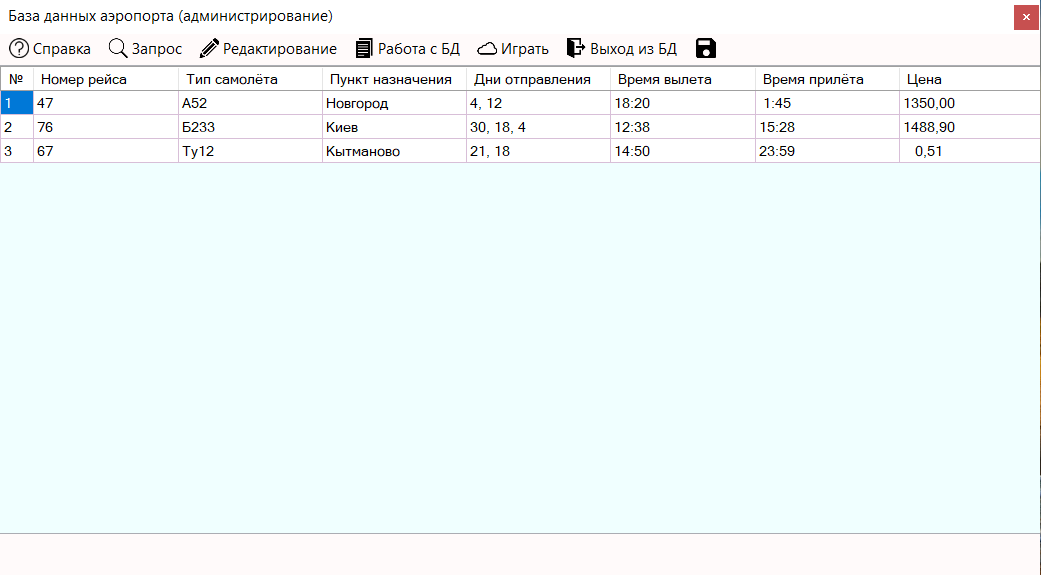
1. Окно авторизации

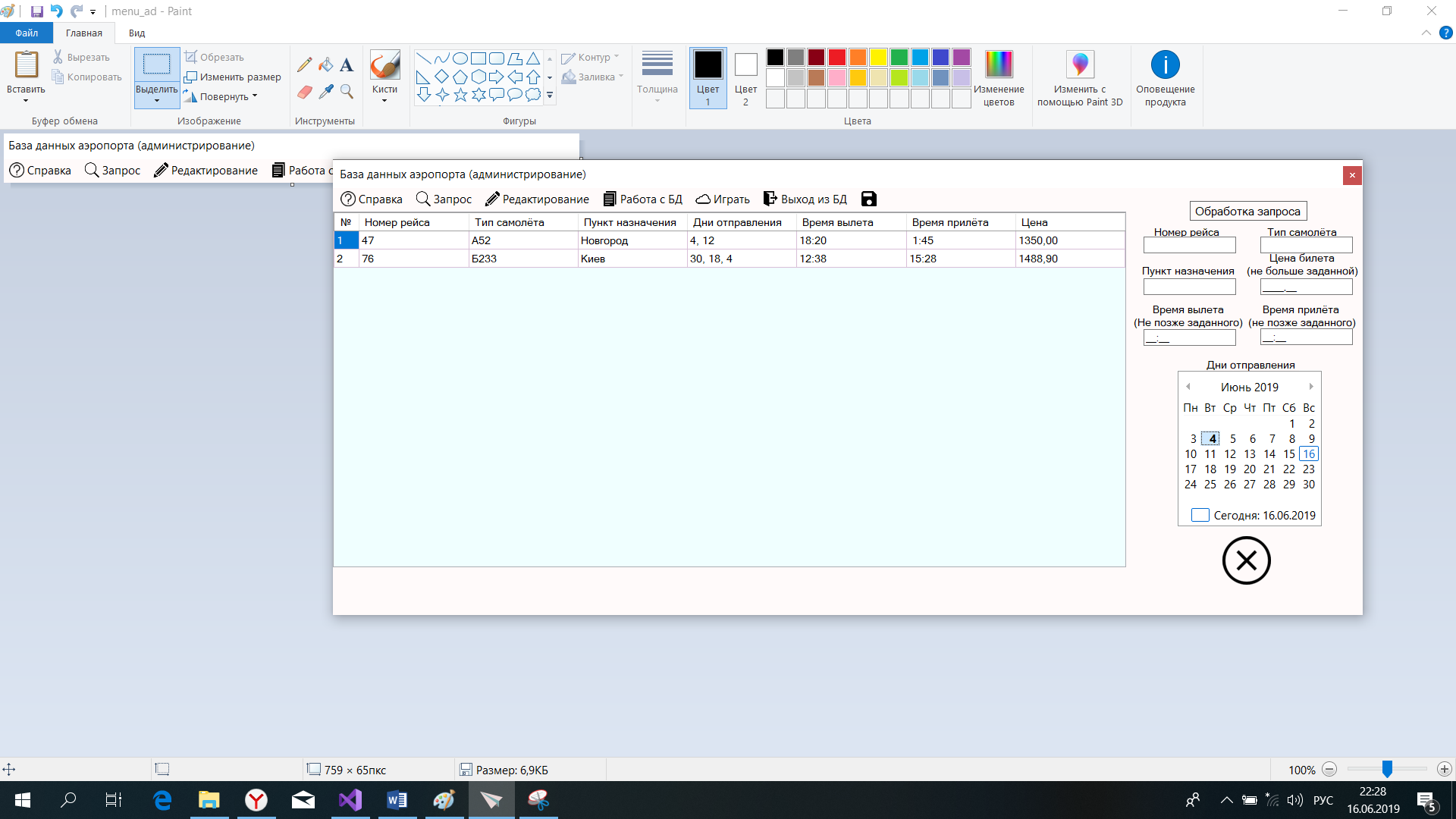
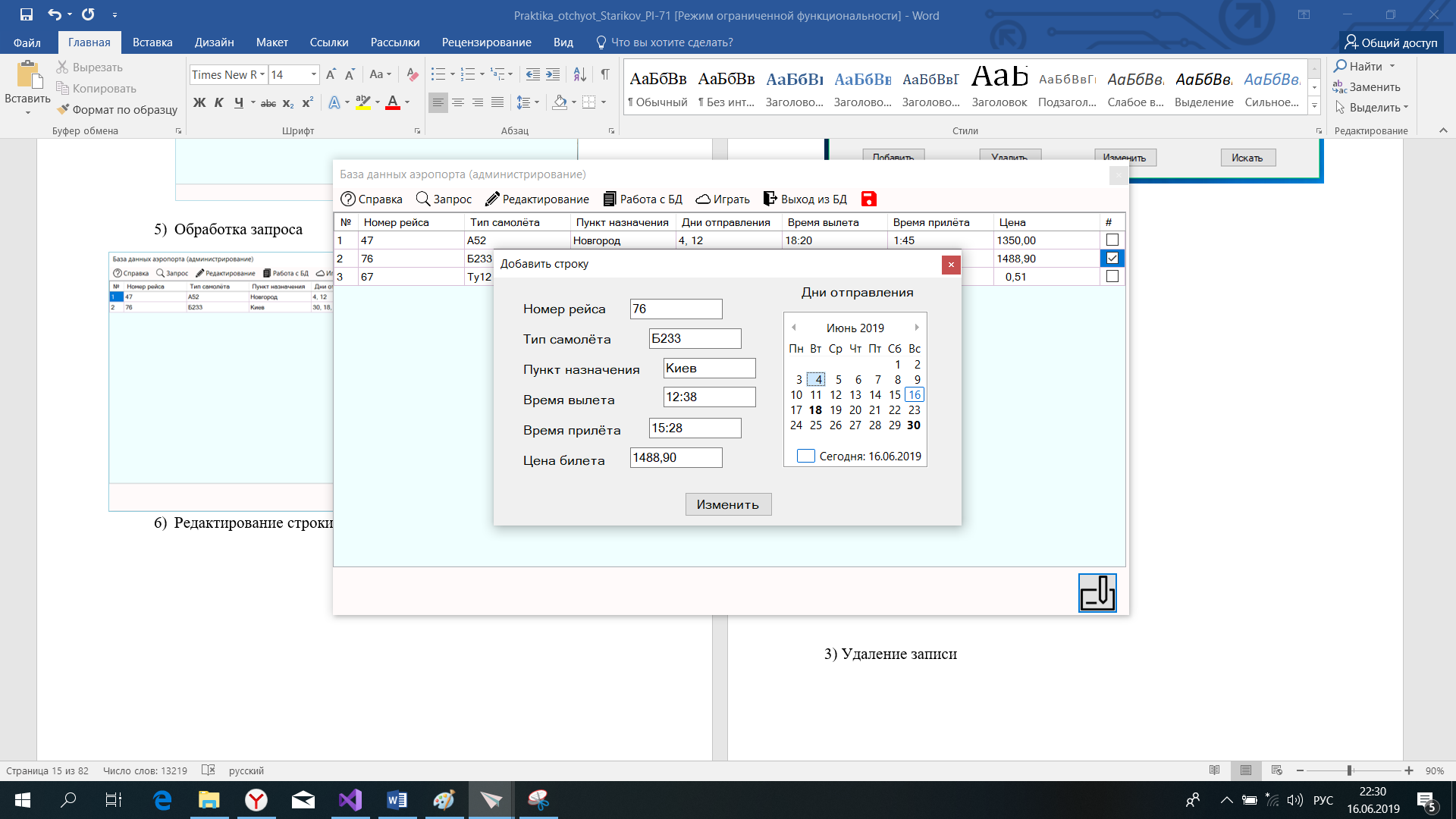


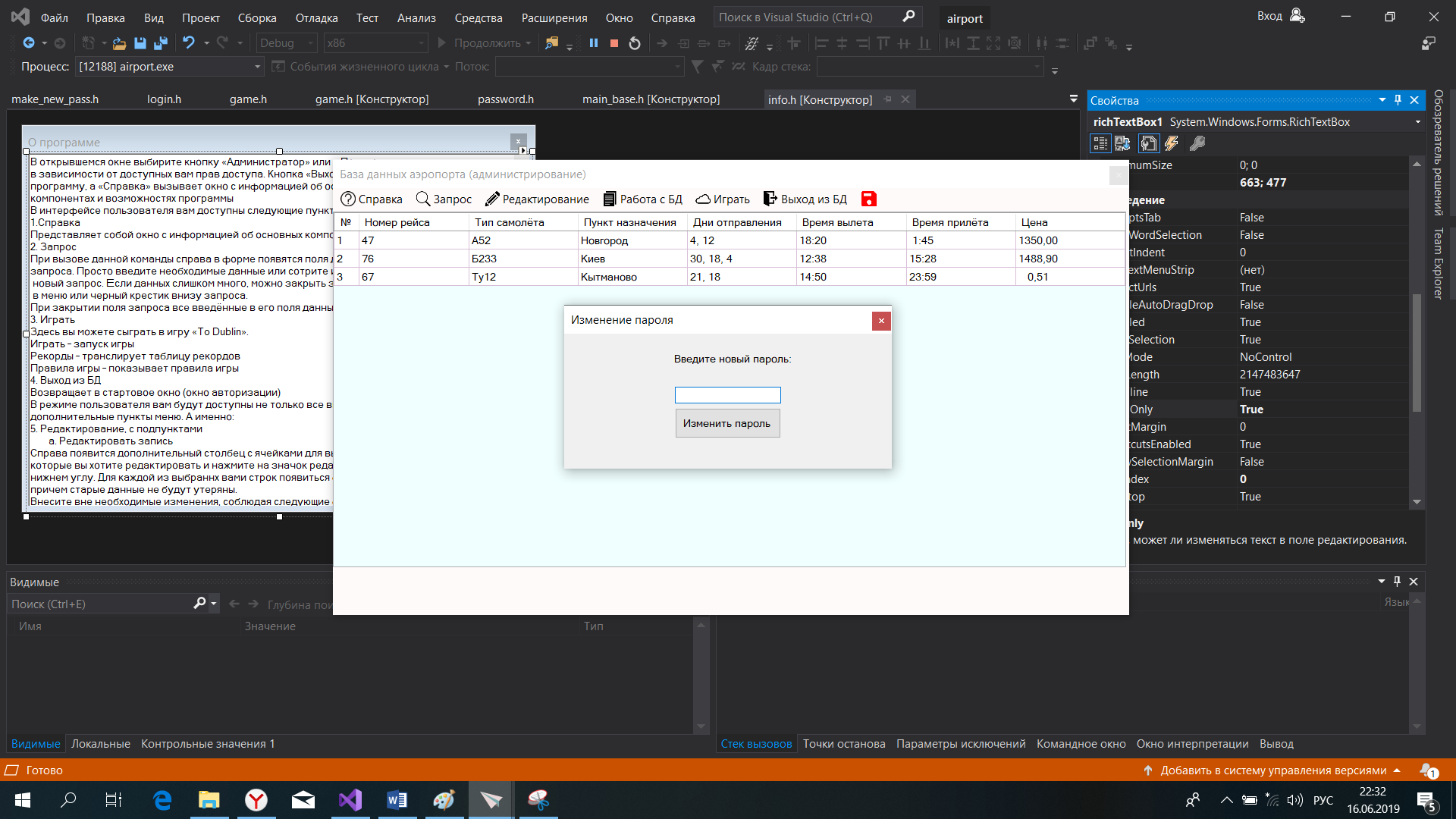
1. Форма для работы с таблицей (интерфейс пользователя)

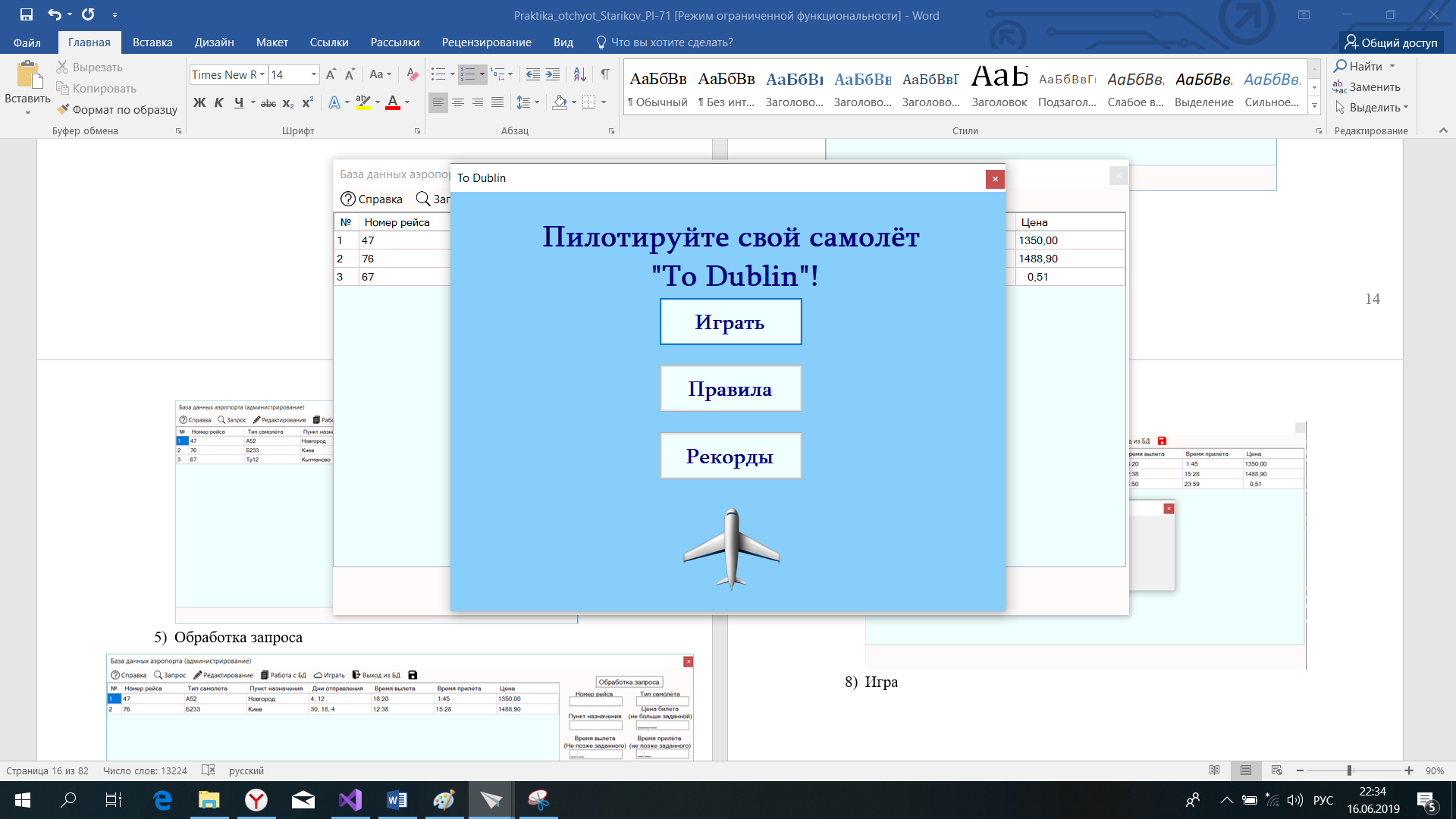
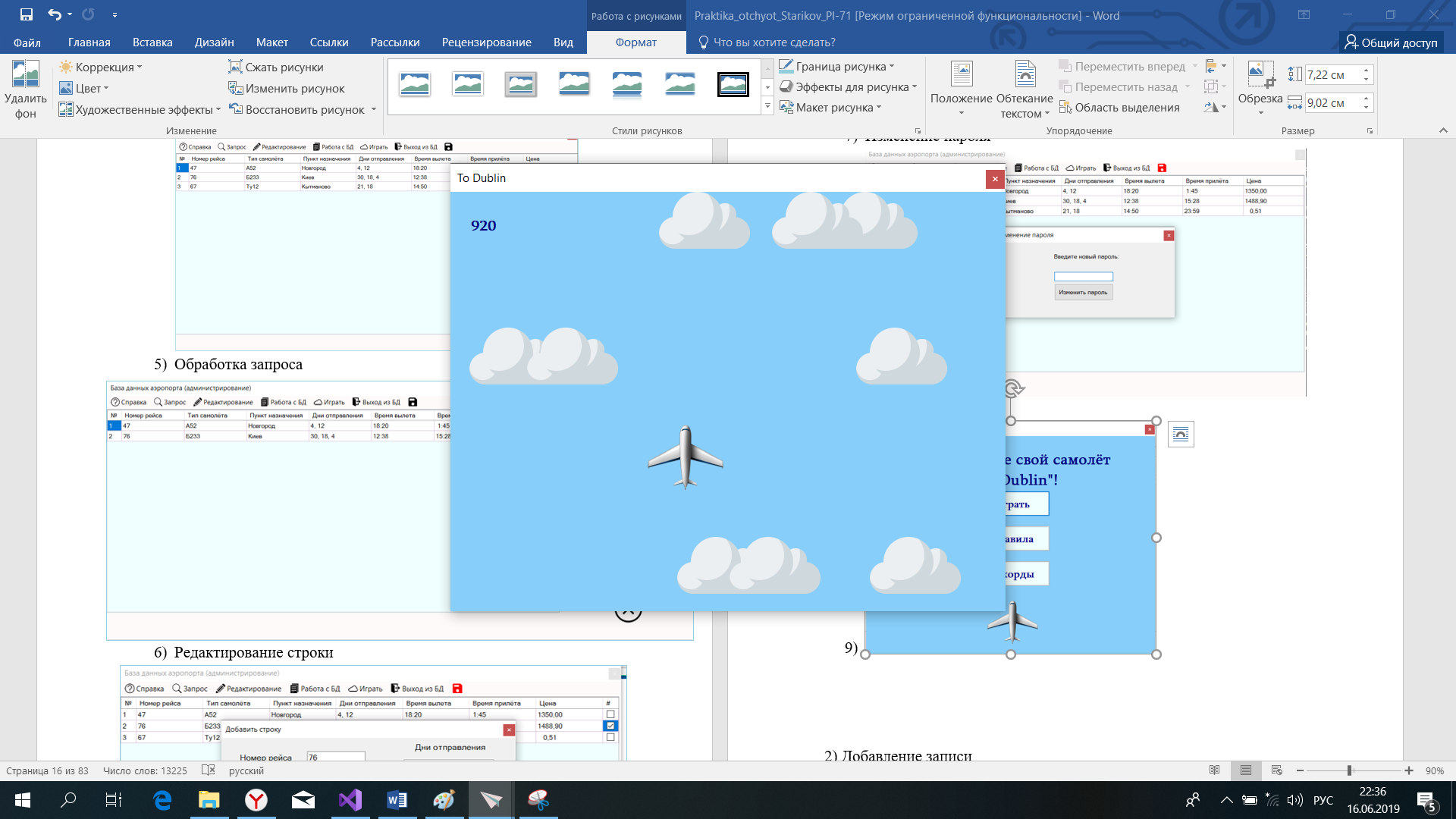


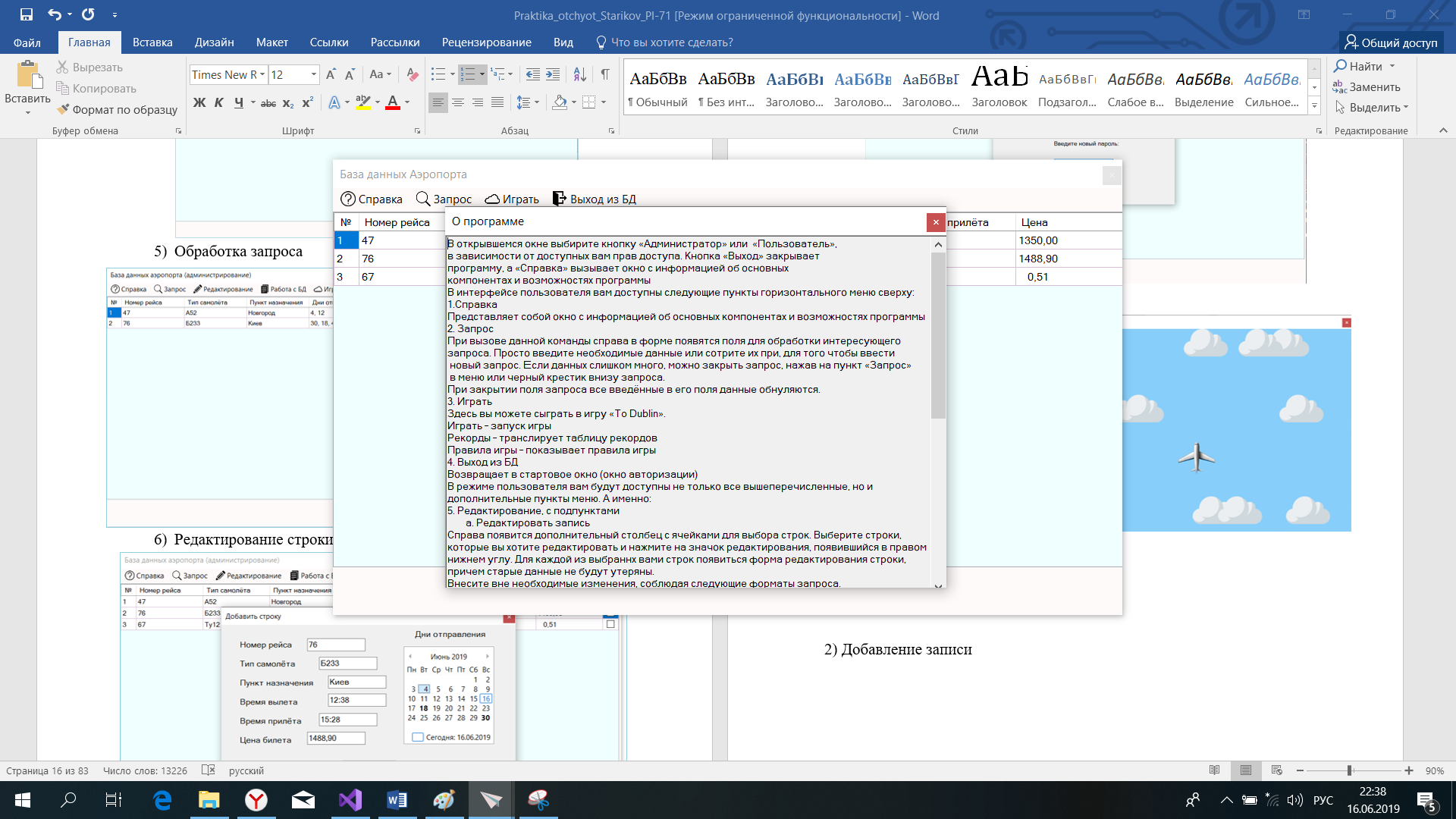
1. Интерфейс администратора



1. Обработка запроса
2. Редактирование строки
3. Изменение пароля



1. Игра
2. Справка



# Приложение Б Исходный код программы

Login.cpp

#include "login.h"

using namespace System;

using namespace System::Windows::Forms;

[STAThreadAttribute]

void main(cli::array<String^>^ args) {

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

airport::login mainForm;

Application::Run(% mainForm);

}

Login.h

#include "main\_base.h"

#include "password.h"

#include "info.h"

#include "make\_new\_pass.h"

#pragma once

namespace airport {

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 login

/// </summary>

public ref class login : public System::Windows::Forms::Form{

public:login(void)

{ InitializeComponent();

//TODO: добавьте код конструктора

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~login(){

if (components)

{delete components;} }

private: System::Windows::Forms::Button^ user\_button;

private: System::Windows::Forms::Button^ admin\_button;

private: System::Windows::Forms::Label^ hello\_text;

private: System::Windows::Forms::Button^ about\_button;

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

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

/// <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(login::typeid));

this->user\_button = (gcnew System::Windows::Forms::Button());

this->admin\_button = (gcnew System::Windows::Forms::Button());

this->hello\_text = (gcnew System::Windows::Forms::Label());

this->about\_button = (gcnew System::Windows::Forms::Button());

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

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

this->SuspendLayout();

// user\_button

this->user\_button->BackColor = System::Drawing::Color::Azure;

this->user\_button->Font = (gcnew System::Drawing::Font(L"Microsoft JhengHei Light", 13.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(0)));

this->user\_button->ForeColor = System::Drawing::SystemColors::ActiveCaptionText;

this->user\_button->Location = System::Drawing::Point(212, 164);

this->user\_button->Name = L"user\_button";

this->user\_button->Size = System::Drawing::Size(232, 59);

this->user\_button->TabIndex = 0;

this->user\_button->Text = L"Пользователь";

this->user\_button->UseVisualStyleBackColor = false;

this->user\_button->Click += gcnew System::EventHandler(this, &login::User\_button\_Click);

// admin\_button

this->admin\_button->BackColor = System::Drawing::Color::Azure;

this->admin\_button->Font = (gcnew System::Drawing::Font(L"Microsoft JhengHei Light", 13.8F, System::Drawing::FontStyle::Regular,

System::Drawing::GraphicsUnit::Point, static\_cast<System::Byte>(0)));

this->admin\_button->ForeColor = System::Drawing::SystemColors::ActiveCaptionText;

this->admin\_button->Location = System::Drawing::Point(212, 229);

this->admin\_button->Name = L"admin\_button";

this->admin\_button->Size = System::Drawing::Size(232, 59);

this->admin\_button->TabIndex = 1;

this->admin\_button->Text = L"Aдминистратор";

this->admin\_button->UseVisualStyleBackColor = false;

this->admin\_button->Click += gcnew System::EventHandler(this, &login::Admin\_button\_Click);

// hello\_text

this->hello\_text->AutoSize = true;

this->hello\_text->BackColor = System::Drawing::Color::Transparent;

this->hello\_text->Font = (gcnew System::Drawing::Font(L"Microsoft JhengHei Light", 10.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(0)));

this->hello\_text->Location = System::Drawing::Point(146, 101);

this->hello\_text->Name = L"hello\_text";

this->hello\_text->Size = System::Drawing::Size(376, 48);

this->hello\_text->TabIndex = 2;

this->hello\_text->Text = L"Для работы с базой данных \r\nвыберите доступные вам права доступа:";

this->hello\_text->TextAlign = System::Drawing::ContentAlignment::TopCenter;

// about\_button

this->about\_button->BackColor = System::Drawing::Color::Transparent;

this->about\_button->Cursor = System::Windows::Forms::Cursors::Hand;

this->about\_button->FlatAppearance->BorderSize = 0;

this->about\_button->FlatStyle = System::Windows::Forms::FlatStyle::Flat;

this->about\_button->Font = (gcnew System::Drawing::Font(L"Microsoft JhengHei Light", 10.2F, System::Drawing::FontStyle::Regular,

System::Drawing::GraphicsUnit::Point, static\_cast<System::Byte>(0)));

this->about\_button->Location = System::Drawing::Point(480, 3);

this->about\_button->Margin = System::Windows::Forms::Padding(0);

this->about\_button->Name = L"about\_button";

this->about\_button->Size = System::Drawing::Size(84, 33);

this->about\_button->TabIndex = 3;

this->about\_button->Text = L"Справка";

this->about\_button->UseVisualStyleBackColor = false;

this->about\_button->Click += gcnew System::EventHandler(this, &login::About\_button\_Click);

// label1

this->label1->AutoSize = true;

this->label1->BackColor = System::Drawing::Color::Transparent;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft JhengHei Light", 22.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(0)));

this->label1->Location = System::Drawing::Point(235, 39);

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

this->label1->Size = System::Drawing::Size(193, 48);

this->label1->TabIndex = 4;

this->label1->Text = L"Аэропорт";

// button1

this->button1->BackColor = System::Drawing::Color::Transparent;

this->button1->Cursor = System::Windows::Forms::Cursors::Hand;

this->button1->FlatAppearance->BorderSize = 0;

this->button1->FlatStyle = System::Windows::Forms::FlatStyle::Flat;

this->button1->Font = (gcnew System::Drawing::Font(L"Microsoft JhengHei Light", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(0)));

this->button1->Location = System::Drawing::Point(564, 3);

this->button1->Margin = System::Windows::Forms::Padding(0);

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

this->button1->Size = System::Drawing::Size(74, 33);

this->button1->TabIndex = 5;

this->button1->Text = L"Выход";

this->button1->UseVisualStyleBackColor = false;

this->button1->Click += gcnew System::EventHandler(this, &login::Button1\_Click\_1);

// login

this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);

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

this->BackColor = System::Drawing::Color::Azure;

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

this->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->ClientSize = System::Drawing::Size(641, 384);

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

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

this->Controls->Add(this->about\_button);

this->Controls->Add(this->hello\_text);

this->Controls->Add(this->admin\_button);

this->Controls->Add(this->user\_button);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::None;

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

this->Name = L"login";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Авторизация";

this->ResumeLayout(false);

this->PerformLayout();}

#pragma endregion

private: System::Void Admin\_button\_Click(System::Object^ sender, System::EventArgs^ e)

{ if (IO::File::Exists("password.txt")) {

password^ pass\_ptr = gcnew password;

pass\_ptr->ShowDialog();

int a = pass\_ptr->is\_admin;

if (a) {

main\_base^ base\_ptr = gcnew main\_base;

base\_ptr->Owner = this;

//задаем текущую форму родительской

base\_ptr->admin = true;

base\_ptr->Show();

this->Hide();}}

else { if (MessageBox::Show("У базы данных отсутствует файл с паролем! \nЗадать новый пароль?", "Изменение пароля", MessageBoxButtons::YesNo, MessageBoxIcon::Question) == System::Windows::Forms::DialogResult::Yes) {

make\_new\_pass^ make\_pass\_ptr = gcnew make\_new\_pass;

make\_pass\_ptr->Show();}}}

private: System::Void User\_button\_Click(System::Object^ sender, System::EventArgs^ e) { main\_base^ user\_form = gcnew main\_base();

user\_form->Text = "База данных Аэропорта";

user\_form->реактироватьToolStripMenuItem->Visible = false;

user\_form->работаСБДToolStripMenuItem->Visible = false;

user\_form->сохранитьToolStripMenuItem->Visible = false;

user\_form->файл\_сохранёнToolStripMenuItem->Visible = false;

user\_form->admin = false;

user\_form->Owner = this;

user\_form->Show();

this->Hide();}private: System::Void Button1\_Click\_1(System::Object^ sender, System::EventArgs^ e) {

Application::Exit();}

private: System::Void About\_button\_Click(System::Object^ sender, System::EventArgs^ e) { info^ about = gcnew info();

about->ShowDialog();}};}

main\_base.h

#include "plus\_string.h"

#include "make\_new\_pass.h"

#include "game.h"

#include "info.h"

using namespace std;

#pragma once

namespace airport {

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 main\_base

/// </summary>

public ref class main\_base : public System::Windows::Forms::Form

{//объявление массива дат

public:cli::array<DateTime>^ current\_times = gcnew array<DateTime>(5);

int count\_date = 0; //индекс для него

bool is\_bold = 0; //флаг: 1 - если дата уже выделена

int index\_bold;

String^ date\_str;

bool day\_correct = 0;

public: System::Windows::Forms::ToolStripMenuItem^ файл\_сохранёнToolStripMenuItem;

System::Windows::Forms::ToolStripMenuItem^ сохранитьToolStripMenuItem;

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

public: bool admin;

void change\_save\_icon(bool admin) {

if (admin){

if (is\_save) {

this->сохранитьToolStripMenuItem->Visible = false;

this->файл\_сохранёнToolStripMenuItem->Visible = true;}

else {

this->сохранитьToolStripMenuItem->Visible = true;

this->файл\_сохранёнToolStripMenuItem->Visible = false;}}}

bool is\_save = true;

main\_base(void)

{ InitializeComponent();

//TODO: добавьте код конструктора

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~main\_base(){

if (components) delete components; }

private: System::Windows::Forms::DataGridView^ main\_table;

System::Windows::Forms::ToolStripMenuItem^ обработатьЗапросToolStripMenuItem;

public: System::Windows::Forms::ToolStripMenuItem^ реактироватьToolStripMenuItem;

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

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

public: System::Windows::Forms::ToolStripMenuItem^ РедактироватьСтрокуToolStripMenuItem;

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

System::Windows::Forms::ToolStripMenuItem^ удалитьСтрокуToolStripMenuItem1;

public: System::Windows::Forms::ToolStripMenuItem^ работаСБДToolStripMenuItem;

System::Windows::Forms::ToolStripMenuItem^ сохранитьИзмененияToolStripMenuItem;

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

System::Windows::Forms::ToolStripMenuItem^ выходИзБДToolStripMenuItem;

System::Windows::Forms::ToolStripMenuItem^ справкаToolStripMenuItem;

System::Windows::Forms::DataGridViewTextBoxColumn^ num\_string;

System::Windows::Forms::DataGridViewTextBoxColumn^ num\_rout;

System::Windows::Forms::DataGridViewTextBoxColumn^ type\_plane;

System::Windows::Forms::DataGridViewTextBoxColumn^ city;

System::Windows::Forms::DataGridViewTextBoxColumn^ days;

System::Windows::Forms::DataGridViewTextBoxColumn^ time\_out;

System::Windows::Forms::DataGridViewTextBoxColumn^ time\_in;

System::Windows::Forms::DataGridViewTextBoxColumn^ price;

System::Windows::Forms::DataGridViewCheckBoxColumn^ deleteColumn;

System::Windows::Forms::Button^ delete\_button;

System::Windows::Forms::Button^ edit\_button;

public: System::Windows::Forms::MaskedTextBox^ price\_Text;

System::Windows::Forms::MaskedTextBox^ time\_in\_Text;

System::Windows::Forms::MaskedTextBox^ time\_out\_Text;

System::Windows::Forms::MaskedTextBox^ city\_Text;

System::Windows::Forms::MaskedTextBox^ type\_plane\_Text;

System::Windows::Forms::MaskedTextBox^ num\_route\_Text;

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

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

public: System::Windows::Forms::MonthCalendar^ monthCalendar1;

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

System::Windows::Forms::PictureBox^ pictureBox1;

System::ComponentModel::IContainer^ components;

/// <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){

System::Windows::Forms::DataGridViewCellStyle^ dataGridViewCellStyle2 = (gcnew System::Windows::Forms::DataGridViewCellStyle());

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(main\_base::typeid));

this->main\_table = (gcnew System::Windows::Forms::DataGridView());

this->num\_string = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->num\_rout = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->type\_plane = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

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

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

this->time\_out = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->time\_in = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

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

this->deleteColumn = (gcnew System::Windows::Forms::DataGridViewCheckBoxColumn());

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

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

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

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

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

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

this->удалитьСтрокуToolStripMenuItem1 = (gcnew System::Windows::Forms::ToolStripMenuItem());

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

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

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

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

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

this->файл\_сохранёнToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

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

this->delete\_button = (gcnew System::Windows::Forms::Button());

this->edit\_button = (gcnew System::Windows::Forms::Button());

this->price\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->time\_in\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->time\_out\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->city\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->type\_plane\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->num\_route\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

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

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

this->monthCalendar1 = (gcnew System::Windows::Forms::MonthCalendar());

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

this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());

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

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

this->menuStrip1->SuspendLayout();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->BeginInit();

this->SuspendLayout();

// main\_table

this->main\_table->AllowUserToAddRows = false;

this->main\_table->AllowUserToDeleteRows = false;

this->main\_table->AutoSizeColumnsMode = System::Windows::Forms::DataGridViewAutoSizeColumnsMode::Fill;

this->main\_table->BackgroundColor = System::Drawing::Color::Azure;

this->main\_table->BorderStyle = System::Windows::Forms::BorderStyle::Fixed3D;

dataGridViewCellStyle2->BackColor = System::Drawing::SystemColors::ButtonHighlight;

dataGridViewCellStyle2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

dataGridViewCellStyle2->ForeColor = System::Drawing::SystemColors::WindowText;

dataGridViewCellStyle2->SelectionBackColor = System::Drawing::Color::SkyBlue;

dataGridViewCellStyle2->SelectionForeColor = System::Drawing::SystemColors::ButtonHighlight;

this->main\_table->ColumnHeadersDefaultCellStyle = dataGridViewCellStyle2;

this->main\_table->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->main\_table->Columns->AddRange(gcnew cli::array< System::Windows::Forms::DataGridViewColumn^ >(9) {

this->num\_string,

this->num\_rout, this->type\_plane, this->city, this->days, this->time\_out, this->time\_in, this->price, this->deleteColumn});

this->main\_table->GridColor = System::Drawing::Color::Thistle;

this->main\_table->Location = System::Drawing::Point(0, 31);

this->main\_table->Margin = System::Windows::Forms::Padding(10);

this->main\_table->Name = L"main\_table";

this->main\_table->RowHeadersBorderStyle = System::Windows::Forms::DataGridViewHeaderBorderStyle::Sunken;

this->main\_table->RowHeadersVisible = false;

this->main\_table->RowHeadersWidth = 51;

this->main\_table->RowTemplate->Height = 24;

this->main\_table->Size = System::Drawing::Size(1046, 469);

this->main\_table->TabIndex = 0;

this->main\_table->CellValueChanged += gcnew System::Windows::Forms::DataGridViewCellEventHandler(this, &main\_base::Main\_table\_CellValueChanged);

// num\_string

this->num\_string->FillWeight = 27.95047F;

this->num\_string->HeaderText = L"№";

this->num\_string->MinimumWidth = 6;

this->num\_string->Name = L"num\_string";

this->num\_string->ReadOnly = true;

this->num\_string->Resizable = System::Windows::Forms::DataGridViewTriState::True;

// num\_rout

this->num\_rout->FillWeight = 120.4532F;

this->num\_rout->HeaderText = L"Номер рейса";

this->num\_rout->MaxInputLength = 10;

this->num\_rout->MinimumWidth = 6;

this->num\_rout->Name = L"num\_rout";

this->num\_rout->ReadOnly = true;

// type\_plane

this->type\_plane->FillWeight = 120.4532F;

this->type\_plane->HeaderText = L"Тип самолёта";

this->type\_plane->MaxInputLength = 20;

this->type\_plane->MinimumWidth = 6;

this->type\_plane->Name = L"type\_plane";

this->type\_plane->ReadOnly = true;

// city

this->city->FillWeight = 120.4532F;

this->city->HeaderText = L"Пункт назначения";

this->city->MaxInputLength = 30;

this->city->MinimumWidth = 6;

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

this->city->ReadOnly = true;

// days

this->days->FillWeight = 120.4532F;

this->days->HeaderText = L"Дни отправления";

this->days->MaxInputLength = 7;

this->days->MinimumWidth = 6;

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

this->days->ReadOnly = true;

// time\_out

this->time\_out->FillWeight = 120.4532F;

this->time\_out->HeaderText = L"Время вылета";

this->time\_out->MaxInputLength = 5;

this->time\_out->MinimumWidth = 6;

this->time\_out->Name = L"time\_out";

this->time\_out->ReadOnly = true;

// time\_in

this->time\_in->FillWeight = 120.4532F;

this->time\_in->HeaderText = L"Время прилёта";

this->time\_in->MaxInputLength = 5;

this->time\_in->MinimumWidth = 6;

this->time\_in->Name = L"time\_in";

this->time\_in->ReadOnly = true;

// price

this->price->FillWeight = 120.4532F;

this->price->HeaderText = L"Цена";

this->price->MaxInputLength = 30;

this->price->MinimumWidth = 6;

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

this->price->ReadOnly = true;

// deleteColumn

this->deleteColumn->FillWeight = 28.877F;

this->deleteColumn->HeaderText = L"#";

this->deleteColumn->MinimumWidth = 6;

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

this->deleteColumn->Visible = false;

// menuStrip1

this->menuStrip1->BackColor = System::Drawing::Color::Snow;

this->menuStrip1->ImageScalingSize = System::Drawing::Size(20, 20);

this->menuStrip1->ImeMode = System::Windows::Forms::ImeMode::NoControl;

this->menuStrip1->Items->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(8) {

this->справкаToolStripMenuItem, this->обработатьЗапросToolStripMenuItem, this->реактироватьToolStripMenuItem, this->работаСБДToolStripMenuItem, this->игратьToolStripMenuItem,

this->выходИзБДToolStripMenuItem, this->файл\_сохранёнToolStripMenuItem, this->сохранитьToolStripMenuItem});

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

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

this->menuStrip1->ShowItemToolTips = true;

this->menuStrip1->Size = System::Drawing::Size(1041, 28);

this->menuStrip1->TabIndex = 1;

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

// справкаToolStripMenuItem

this->справкаToolStripMenuItem->AutoToolTip = true;

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

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

this->справкаToolStripMenuItem->Size = System::Drawing::Size(101, 24);

this->справкаToolStripMenuItem->Text = L"Справка";

this->справкаToolStripMenuItem->ToolTipText = L"Информация о программе";this->справкаToolStripMenuItem->Click += gcnew System::EventHandler(this, &main\_base::СправкаToolStripMenuItem\_Click);

// обработатьЗапросToolStripMenuItem

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

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

this->обработатьЗапросToolStripMenuItem->Size = System::Drawing::Size(93, 24);

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

this->обработатьЗапросToolStripMenuItem->ToolTipText = L"Обработка универсального запроса по заданным параметрам.";

this->обработатьЗапросToolStripMenuItem->Click += gcnew System::EventHandler(this, &main\_base::ОбработатьЗапросToolStripMenuItem\_Click);

// реактироватьToolStripMenuItem

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

this->РедактироватьСтрокуToolStripMenuItem,

this->добавитьСтрокуToolStripMenuItem, this->удалитьСтрокуToolStripMenuItem1});

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

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

this->реактироватьToolStripMenuItem->Size = System::Drawing::Size(157, 24);

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

// РедактироватьСтрокуToolStripMenuItem

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

this->РедактироватьСтрокуToolStripMenuItem->Size = System::Drawing::Size(246, 26);

this->РедактироватьСтрокуToolStripMenuItem->Text = L"Редактировать запись";

this->РедактироватьСтрокуToolStripMenuItem->Click += gcnew System::EventHandler(this, &main\_base::РедактироватьСтрокуToolStripMenuItem\_Click);

// добавитьСтрокуToolStripMenuItem

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

this->добавитьСтрокуToolStripMenuItem->Size = System::Drawing::Size(246, 26);

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

this->добавитьСтрокуToolStripMenuItem->Click += gcnew System::EventHandler(this, &main\_base::ДобавитьСтрокуToolStripMenuItem\_Click);

// удалитьСтрокуToolStripMenuItem1

this->удалитьСтрокуToolStripMenuItem1->Name = L"удалитьСтрокуToolStripMenuItem1";

this->удалитьСтрокуToolStripMenuItem1->Size = System::Drawing::Size(246, 26);

this->удалитьСтрокуToolStripMenuItem1->Text = L"Удалить запись";

this->удалитьСтрокуToolStripMenuItem1->Click += gcnew System::EventHandler(this, &main\_base::УдалитьСтрокуToolStripMenuItem1\_Click);

// работаСБДToolStripMenuItem

this->работаСБДToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(2) {

this->сохранитьИзмененияToolStripMenuItem,

this->изменитьПарольToolStripMenuItem});

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

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

this->работаСБДToolStripMenuItem->Size = System::Drawing::Size(125, 24);

this->работаСБДToolStripMenuItem->Text = L"Работа с БД";

// сохранитьИзмененияToolStripMenuItem

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

this->сохранитьИзмененияToolStripMenuItem->Size = System::Drawing::Size(248, 26);

this->сохранитьИзмененияToolStripMenuItem->Text = L"Сохранить изменения";

this->сохранитьИзмененияToolStripMenuItem->Click += gcnew System::EventHandler(this, &main\_base::СохранитьИзмененияToolStripMenuItem\_Click);

// изменитьПарольToolStripMenuItem

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

this->изменитьПарольToolStripMenuItem->Size = System::Drawing::Size(248, 26);

this->изменитьПарольToolStripMenuItem->Text = L"Изменить пароль";

this->изменитьПарольToolStripMenuItem->Click += gcnew System::EventHandler(this, &main\_base::ИзменитьПарольToolStripMenuItem\_Click);

// игратьToolStripMenuItem

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

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

this->игратьToolStripMenuItem->Size = System::Drawing::Size(91, 24);

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

this->игратьToolStripMenuItem->ToolTipText = L"Пилотируйте свой самолёт!";

this->игратьToolStripMenuItem->Click += gcnew System::EventHandler(this, &main\_base::ИгратьToolStripMenuItem\_Click);

// выходИзБДToolStripMenuItem

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

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

this->выходИзБДToolStripMenuItem->Size = System::Drawing::Size(130, 24);

this->выходИзБДToolStripMenuItem->Text = L"Выход из БД";

this->выходИзБДToolStripMenuItem->Click += gcnew System::EventHandler(this, &main\_base::ВыходИзБДToolStripMenuItem\_Click);

// файл\_сохранёнToolStripMenuItem

this->файл\_сохранёнToolStripMenuItem->AutoToolTip = true;

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

this->файл\_сохранёнToolStripMenuItem->Name = L"файл\_сохранёнToolStripMenuItem";

this->файл\_сохранёнToolStripMenuItem->Size = System::Drawing::Size(34, 24);

this->файл\_сохранёнToolStripMenuItem->ToolTipText = L"Файл сохранён";

// сохранитьToolStripMenuItem

this->сохранитьToolStripMenuItem->AutoToolTip = true;

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

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

this->сохранитьToolStripMenuItem->Size = System::Drawing::Size(34, 24);

this->сохранитьToolStripMenuItem->ToolTipText = L"Сохранить файл";

this->сохранитьToolStripMenuItem->Click += gcnew System::EventHandler(this, &main\_base::СохранитьToolStripMenuItem\_Click\_1);

// delete\_button

this->delete\_button->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"delete\_button.BackgroundImage")));

this->delete\_button->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Center;

this->delete\_button->Location = System::Drawing::Point(983, 503);

this->delete\_button->Name = L"delete\_button";

this->delete\_button->Size = System::Drawing::Size(53, 55);

this->delete\_button->TabIndex = 2;

this->delete\_button->UseVisualStyleBackColor = true;

this->delete\_button->Visible = false;

this->delete\_button->Click += gcnew System::EventHandler(this, &main\_base::Delete\_button\_Click);

// edit\_button

this->edit\_button->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"edit\_button.BackgroundImage")));

this->edit\_button->BackgroundImageLayout = System::Windows::Forms::ImageLayout::Stretch;

this->edit\_button->Location = System::Drawing::Point(982, 507);

this->edit\_button->Name = L"edit\_button";

this->edit\_button->Size = System::Drawing::Size(53, 54);

this->edit\_button->TabIndex = 3;

this->edit\_button->UseVisualStyleBackColor = true;

this->edit\_button->Visible = false;

this->edit\_button->Click += gcnew System::EventHandler(this, &main\_base::Edit\_button\_Click);

// price\_Text

this->price\_Text->BeepOnError = true;

this->price\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->price\_Text->Location = System::Drawing::Point(1223, 119);

this->price\_Text->Mask = L"9990.00";

this->price\_Text->Name = L"price\_Text";

this->price\_Text->Size = System::Drawing::Size(122, 22);

this->price\_Text->TabIndex = 14;

this->price\_Text->TextChanged += gcnew System::EventHandler(this, &main\_base::Price\_Text\_TextChanged);

// time\_in\_Text

this->time\_in\_Text->BeepOnError = true;

this->time\_in\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->time\_in\_Text->Location = System::Drawing::Point(1223, 185);

this->time\_in\_Text->Mask = L"90:00";

this->time\_in\_Text->Name = L"time\_in\_Text";

this->time\_in\_Text->Size = System::Drawing::Size(122, 22);

this->time\_in\_Text->TabIndex = 13;

this->time\_in\_Text->ValidatingType = System::DateTime::typeid;

this->time\_in\_Text->TextChanged += gcnew System::EventHandler(this, &main\_base::Time\_in\_Text\_TextChanged);

// time\_out\_Text

this->time\_out\_Text->BeepOnError = true;

this->time\_out\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->time\_out\_Text->Location = System::Drawing::Point(1069, 186);

this->time\_out\_Text->Mask = L"90:00";

this->time\_out\_Text->Name = L"time\_out\_Text";

this->time\_out\_Text->Size = System::Drawing::Size(122, 22);

this->time\_out\_Text->TabIndex = 12;

this->time\_out\_Text->ValidatingType = System::DateTime::typeid;

this->time\_out\_Text->TextChanged += gcnew System::EventHandler(this, &main\_base::Time\_out\_Text\_TextChanged);

// city\_Text

this->city\_Text->BeepOnError = true;

this->city\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->city\_Text->Location = System::Drawing::Point(1069, 119);

this->city\_Text->Mask = L"L\?\?\?\?\?\?\?\?\?";

this->city\_Text->Name = L"city\_Text";

this->city\_Text->PromptChar = ' ';

this->city\_Text->Size = System::Drawing::Size(122, 22);

this->city\_Text->TabIndex = 11;

this->city\_Text->Click += gcnew System::EventHandler(this, &main\_base::City\_Text\_Click);

this->city\_Text->TextChanged += gcnew System::EventHandler(this, &main\_base::City\_Text\_TextChanged);

// type\_plane\_Text

this->type\_plane\_Text->BeepOnError = true;

this->type\_plane\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->type\_plane\_Text->Location = System::Drawing::Point(1223, 64);

this->type\_plane\_Text->Mask = L"Aaaaa";

this->type\_plane\_Text->Name = L"type\_plane\_Text";

this->type\_plane\_Text->PromptChar = ' ';

this->type\_plane\_Text->Size = System::Drawing::Size(122, 22);

this->type\_plane\_Text->TabIndex = 10;

this->type\_plane\_Text->Click += gcnew System::EventHandler(this, &main\_base::Type\_plane\_Text\_Click);

this->type\_plane\_Text->TextChanged += gcnew System::EventHandler(this, &main\_base::Type\_plane\_Text\_TextChanged);

// num\_route\_Text

this->num\_route\_Text->AllowDrop = true;

this->num\_route\_Text->BeepOnError = true;

this->num\_route\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->num\_route\_Text->Location = System::Drawing::Point(1069, 64);

this->num\_route\_Text->Mask = L"999999";

this->num\_route\_Text->Name = L"num\_route\_Text";

this->num\_route\_Text->PromptChar = ' ';

this->num\_route\_Text->Size = System::Drawing::Size(122, 22);

this->num\_route\_Text->TabIndex = 9;

this->num\_route\_Text->ValidatingType = System::Int32::typeid;

this->num\_route\_Text->Click += gcnew System::EventHandler(this, &main\_base::Num\_route\_Text\_Click);

this->num\_route\_Text->TextChanged += gcnew System::EventHandler(this, &main\_base::Num\_route\_Text\_TextChanged);

// label1

this->label1->AutoSize = true;

this->label1->BackColor = System::Drawing::Color::Transparent;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->label1->Location = System::Drawing::Point(1053, 49);

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

this->label1->Size = System::Drawing::Size(152, 170);

this->label1->TabIndex = 8;

this->label1->Text = L"Номер рейса \r\n\r\n\r\nПункт назначения\r\n\r\n\r\nВремя вылета\r\n(Не позже заданного)\r\n\r\n\r\n";

this->label1->TextAlign = System::Drawing::ContentAlignment::TopCenter;

// label2

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->label2->Location = System::Drawing::Point(1149, 224);

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

this->label2->Size = System::Drawing::Size(125, 17);

this->label2->TabIndex = 16;

this->label2->Text = L"Дни отправления";

// monthCalendar1

this->monthCalendar1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->monthCalendar1->Location = System::Drawing::Point(1113, 240);

this->monthCalendar1->MaxDate = System::DateTime(2019, 6, 30, 0, 0, 0, 0);

this->monthCalendar1->MaxSelectionCount = 1;

this->monthCalendar1->MinDate = System::DateTime(2019, 6, 1, 0, 0, 0, 0);

this->monthCalendar1->Name = L"monthCalendar1";

this->monthCalendar1->TabIndex = 15;

this->monthCalendar1->DateSelected += gcnew System::Windows::Forms::DateRangeEventHandler(this, &main\_base::MonthCalendar1\_DateSelected);

// label3

this->label3->AutoSize = true;

this->label3->BackColor = System::Drawing::Color::Transparent;

this->label3->Location = System::Drawing::Point(1202, 49);

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

this->label3->Size = System::Drawing::Size(155, 170);

this->label3->TabIndex = 17;

this->label3->Text = L"Тип самолёта\r\n\r\nЦена билета\r\n(не больше заданной)\r\n\r\n\r\nВремя прилёта \r\n(не позже "

L"заданного)\r\n\r\n\r\n";

this->label3->TextAlign = System::Drawing::ContentAlignment::TopCenter;

// pictureBox1

this->pictureBox1->Image = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"pictureBox1.Image")));

this->pictureBox1->Location = System::Drawing::Point(1173, 459);

this->pictureBox1->Name = L"pictureBox1";

this->pictureBox1->Size = System::Drawing::Size(69, 67);

this->pictureBox1->TabIndex = 18;

this->pictureBox1->TabStop = false;

this->pictureBox1->Click += gcnew System::EventHandler(this, &main\_base::PictureBox1\_Click);

// label4

this->label4->AutoSize = true;

this->label4->BackColor = System::Drawing::Color::Transparent;

this->label4->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->label4->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 9, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->label4->Location = System::Drawing::Point(1130, 17);

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

this->label4->Padding = System::Windows::Forms::Padding(3);

this->label4->Size = System::Drawing::Size(155, 26);

this->label4->TabIndex = 19;

this->label4->Text = L"Обработка запроса";

// main\_base

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::None;

this->BackColor = System::Drawing::Color::Snow;

this->ClientSize = System::Drawing::Size(1041, 563);

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

this->Controls->Add(this->pictureBox1);

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

this->Controls->Add(this->monthCalendar1);

this->Controls->Add(this->price\_Text);

this->Controls->Add(this->time\_in\_Text);

this->Controls->Add(this->time\_out\_Text);

this->Controls->Add(this->city\_Text);

this->Controls->Add(this->type\_plane\_Text);

this->Controls->Add(this->num\_route\_Text);

this->Controls->Add(this->edit\_button);

this->Controls->Add(this->delete\_button);

this->Controls->Add(this->main\_table);

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

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

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

this->DoubleBuffered = true;

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

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

this->MainMenuStrip = this->menuStrip1;

this->MaximizeBox = false;

this->MinimizeBox = false;

this->Name = L"main\_base";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"База данных аэропорта (администрирование)";

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

this->Load += gcnew System::EventHandler(this, &main\_base::Main\_base\_Load);

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

this->menuStrip1->ResumeLayout(false);

this->menuStrip1->PerformLayout();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

void fill\_table(DataGridView^ table, int num\_string) {

StreamReader^ str = gcnew StreamReader("base.txt");

String^ buf\_string;

for (int i = 0; num\_string ? i < num\_string : !str->EndOfStream; i++) {

table->Rows->Add();

buf\_string = str->ReadLine();

add\_string(table, i, buf\_string); }

is\_save = true;

change\_save\_icon(admin);

str->Close();};

void add\_string(DataGridView^ table, int num\_string, String^ str) {

cli::array<String^>^ cells\_fill;

cells\_fill = str->Split('|');

table->Rows[num\_string]->Cells[0]->Value = num\_string + 1; //заполняем первый столбец (номер строки)

for (int j = 1; j < 8; j++) {

table->Rows[num\_string]->Cells[j]->Value = cells\_fill[j - 1]; }}

#pragma endregion

private: System::Void Main\_base\_Load(System::Object^ sender, System::EventArgs^ e) {

if (IO::File::Exists("base.txt")) { //проверка на существование файла

fill\_table(main\_table, 0);

change\_save\_icon(admin);}

else {

//создание нового файла

StreamWriter^ new\_file = gcnew StreamWriter("base.txt");

delete new\_file;

fill\_table(main\_table, 0);}}

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

if (is\_save == false)

switch (MessageBox::Show("При выходе из базы данных все несохраненные данные будут потеряны.\nСохранить изменения?", "Предупреждение", MessageBoxButtons::YesNoCancel, MessageBoxIcon::Warning)) {

case System::Windows::Forms::DialogResult::Yes: СохранитьИзмененияToolStripMenuItem\_Click(sender, e); break;

case System::Windows::Forms::DialogResult::No: break;

case System::Windows::Forms::DialogResult::Cancel: e->Cancel = true; return; }

Owner->Show();}

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

game^ start = gcnew game();

start->Show();}

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

plus\_string^ add\_s = gcnew plus\_string();

add\_s->ShowDialog();

if (add\_s->clear\_str != String::Empty) {

main\_table->Rows->Add();

add\_string(main\_table, main\_table->RowCount - 1, add\_s->clear\_str);}}

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

if (is\_save == false)

switch (MessageBox::Show("При выходе из базы данных все несохраненные данные будут потеряны.\nСохранить изменения?", "Предупреждение", MessageBoxButtons::YesNoCancel, MessageBoxIcon::Warning)) {

case System::Windows::Forms::DialogResult::Yes: СохранитьИзмененияToolStripMenuItem\_Click(sender, e); break;

case System::Windows::Forms::DialogResult::No:break;

case System::Windows::Forms::DialogResult::Cancel: return; }

Owner->Show();

this->Hide();

}

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

make\_new\_pass^ pass = gcnew make\_new\_pass();

pass->ShowDialog();}

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

StreamWriter^ ptr = gcnew StreamWriter("base.txt"); //открываем файл для записи правды

for (int i = 0; i < main\_table->RowCount; i++) {

for (int j = 1; j < 8; j++) {

ptr->Write(main\_table->Rows[i]->Cells[j]->Value + "|"); }

ptr->WriteLine();}

ptr->Close();

is\_save = true;

change\_save\_icon(admin);

MessageBox::Show("Файл сохранён", "Успех", MessageBoxButtons::OK, MessageBoxIcon::Information);}

private: System::Void Main\_table\_CellValueChanged(System::Object^ sender, System::Windows::Forms::DataGridViewCellEventArgs^ e) {

is\_save = false;

change\_save\_icon(admin);}

private: System::Void УдалитьСтрокуToolStripMenuItem1\_Click(System::Object^ sender, System::EventArgs^ e) {

main\_table->Columns[8]->Visible = true;

delete\_button->Visible = true;}

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

System::Windows::Forms::DialogResult what = MessageBox::Show("Данные будут удалены.\nВы уверены?", "Предупреждение", MessageBoxButtons::YesNo, MessageBoxIcon::Question);

if (what == System::Windows::Forms::DialogResult::No)

return;

else {

for (int i = 0; i < main\_table->RowCount; i++) {

if (main\_table->Rows[i]->Cells[8]->Value) main\_table->Rows->RemoveAt(i);

main\_table->Columns[8]->Visible = false;

delete\_button->Visible = false;}}}

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

main\_table->Columns[8]->Visible = true;

edit\_button->Visible = true;}

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

for (int i = 0; i < main\_table->RowCount; i++)

if (main\_table->Rows[i]->Cells[8]->FormattedValue->Equals(true)) {

plus\_string^ sinkin = gcnew plus\_string();

sinkin->Name = "Редактировать строку";

sinkin->create\_button->Text = "Изменить";

sinkin->num\_route\_Text->Text = main\_table->Rows[i]->Cells[1]->Value + "";

sinkin->type\_plane\_Text->Text = main\_table->Rows[i]->Cells[2]->Value + "";

sinkin->city\_Text->Text = main\_table->Rows[i]->Cells[3]->Value + "";

sinkin->time\_out\_Text->Text = main\_table->Rows[i]->Cells[5]->Value + "";

sinkin->time\_in\_Text->Text = main\_table->Rows[i]->Cells[6]->Value + "";

sinkin->price\_Text->Text = main\_table->Rows[i]->Cells[7]->Value + "";

DateTime^ now = gcnew DateTime();

now = DateTime::Now;

String^ dateString;

array<String^>^ for\_split = main\_table->Rows[i]->Cells[4]->Value->ToString()->Split(',');

for (int j = 0; j < for\_split->Length; j++) {

dateString = for\_split[j]->Trim() + "." + now->Month + "." + now->Year + " 00:00:00";

sinkin->current\_times[j] = DateTime::Parse(dateString); //присваивает значение выделенной ячейки переменной

sinkin->monthCalendar1->AddBoldedDate(sinkin->current\_times[j]); //делает число жирным

sinkin->count\_date++;

}

sinkin->monthCalendar1->UpdateBoldedDates();

sinkin->ShowDialog();

if (sinkin->clear\_str != String::Empty) {

add\_string(main\_table, i, sinkin->clear\_str);

}

delete sinkin; }

main\_table->Columns[8]->Visible = false;

edit\_button->Visible = false;} ///

///обработка запроса

///

private: System::Void Num\_route\_Text\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Check();}

void Check() {

for (int i = 0; i < main\_table->RowCount; i++) {

bool is\_right = true;

if (num\_route\_Text->Text != String::Empty)

if (Convert::ToInt32(main\_table->Rows[i]->Cells[1]->Value) != Convert::ToInt32(num\_route\_Text->Text))

is\_right = false;

if (type\_plane\_Text->Text != String::Empty)

if (Convert::ToString(main\_table->Rows[i]->Cells[2]->Value)->IndexOf(type\_plane\_Text->Text) < 0)

is\_right = false;

if (city\_Text->Text != String::Empty)

if (Convert::ToString(main\_table->Rows[i]->Cells[3]->Value)->IndexOf(city\_Text->Text) < 0)

is\_right = false;

if (time\_in\_Text->Text != " :") {

array<String^>^ input\_str = time\_in\_Text->Text->Split(':');

array<String^>^ table\_str = Convert::ToString(main\_table->Rows[i]->Cells[6]->Value)->Split(':');

if (!String::IsNullOrWhiteSpace(input\_str[0]))

if (Convert::ToInt32(input\_str[0]) < Convert::ToInt32(table\_str[0]))

is\_right = false;

else if (input\_str[1] != String::Empty)

if (Convert::ToInt32(input\_str[1]) < Convert::ToInt32(table\_str[1]))

is\_right = false;

}

if (time\_out\_Text->Text != " :") {

array<String^>^ input\_str\_in = time\_out\_Text->Text->Split(':');

array<String^>^ table\_str = Convert::ToString(main\_table->Rows[i]->Cells[5]->Value)->Split(':');

if (!String::IsNullOrWhiteSpace(input\_str\_in[0]))

if (Convert::ToInt32(input\_str\_in[0]) < Convert::ToInt32(table\_str[0]))

is\_right = false;

else if (input\_str\_in[1] != String::Empty)

if (Convert::ToInt32(input\_str\_in[1]) < Convert::ToInt32(table\_str[1]))

is\_right = false;

}

if (price\_Text->Text->Trim() != ",") {

if (Convert::ToDouble("0" + price\_Text->Text->Trim()) < Convert::ToDouble(main\_table->Rows[i]->Cells[7]->Value))

is\_right = false;

}

if (count\_date) {

array<String^>^ tab\_date = Convert::ToString(main\_table->Rows[i]->Cells[4]->Value)->Split(',');

int its;

for (int j = 0; j < count\_date; j++) {

its = 0;

for (int k = 0; k < tab\_date->Length; k++) {

if (Convert::ToInt32(tab\_date[k]) == current\_times[j].Day)

its = 1;

}

if (its == 0)

is\_right = false;

}

}

if (is\_right)

main\_table->Rows[i]->Visible = true;

else main\_table->Rows[i]->Visible = false;

}

}

private: System::Void MonthCalendar1\_DateSelected(System::Object^ sender, System::Windows::Forms::DateRangeEventArgs^ e) {

for (int i = 0; i < count\_date; i++)

if (DateTime::Equals(current\_times[i], monthCalendar1->SelectionStart)) {

is\_bold = 1;

index\_bold = i; }

if (is\_bold) {

monthCalendar1->RemoveBoldedDate(monthCalendar1->SelectionStart);

for (int i = index\_bold; i < count\_date - 1; i++)

current\_times[i] = current\_times[i + 1];

count\_date--;

is\_bold = 0;}

else if (count\_date < 5)

{

current\_times[count\_date] = monthCalendar1->SelectionStart; //присваивает значение выделенной ячейки переменной

count\_date++;

monthCalendar1->AddBoldedDate(monthCalendar1->SelectionStart); //делает число жирным}

else {

MessageBox::Show("Нельзя выбрать больше 5 дней отправления", "Ошибка ввода", MessageBoxButtons::OK, MessageBoxIcon::Hand);}

monthCalendar1->UpdateBoldedDates(); //обновляет календарь

Check();};

private: System::Void Type\_plane\_Text\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Check();}

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

Check();}

private: System::Void Time\_out\_Text\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Check();}

private: System::Void Time\_in\_Text\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

Check();}

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

Check();}

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

if (Width < 1376)

Width = 1376;

else PictureBox1\_Click(sender, e);}

private: System::Void PictureBox1\_Click(System::Object^ sender, System::EventArgs^ e) {

Width = 1068;

num\_route\_Text->Text = "";

type\_plane\_Text->Text = "";

city\_Text->Text = "";

time\_out\_Text->Text = "";

time\_in\_Text->Text = "";

price\_Text->Text = "";

for (int i = 0; i < main\_table->RowCount; i++) {

main\_table->Rows[i]->Visible = true;}}

//ставим курсор сразу за концом текста

private: System::Void Num\_route\_Text\_Click(System::Object^ sender, System::EventArgs^ e) {

num\_route\_Text->SelectionStart = num\_route\_Text->Text->Length;}

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

city\_Text->SelectionStart = city\_Text->Text->Length;}

private: System::Void Type\_plane\_Text\_Click(System::Object^ sender, System::EventArgs^ e) {

type\_plane\_Text->SelectionStart = type\_plane\_Text->Text->Length;}

private: System::Void СохранитьToolStripMenuItem\_Click\_1(System::Object^ sender, System::EventArgs^ e) {

СохранитьИзмененияToolStripMenuItem\_Click(sender, e);}

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

info^ about = gcnew info();

about->ShowDialog();

}

};

}

Make\_new\_pass.h

#pragma once

namespace airport {

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 make\_new\_pass

/// </summary>

public ref class make\_new\_pass : public System::Windows::Forms::Form

{

public:

int score;

String^ pass\_input;

bool is\_pass = true;

public: System::Windows::Forms::MaskedTextBox^ maskedTextBox1;

public:

String^ name\_file = "password.txt";

make\_new\_pass(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~make\_new\_pass()

{

if (components)

{

delete components;

}}

public: System::Windows::Forms::TextBox^ new\_pass\_textBox;

System::Windows::Forms::Button^ save\_pass\_button;

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

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

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(make\_new\_pass::typeid));

this->new\_pass\_textBox = (gcnew System::Windows::Forms::TextBox());

this->save\_pass\_button = (gcnew System::Windows::Forms::Button());

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

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

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

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

this->SuspendLayout();

// new\_pass\_textBox

this->new\_pass\_textBox->Location = System::Drawing::Point(146, 70);

this->new\_pass\_textBox->MaxLength = 25;

this->new\_pass\_textBox->MinimumSize = System::Drawing::Size(140, 30);

this->new\_pass\_textBox->Name = L"new\_pass\_textBox";

this->new\_pass\_textBox->PasswordChar = '\*';

this->new\_pass\_textBox->Size = System::Drawing::Size(140, 22);

this->new\_pass\_textBox->TabIndex = 2;

// save\_pass\_button

this->save\_pass\_button->Location = System::Drawing::Point(146, 98);

this->save\_pass\_button->Name = L"save\_pass\_button";

this->save\_pass\_button->Size = System::Drawing::Size(140, 40);

this->save\_pass\_button->TabIndex = 3;

this->save\_pass\_button->Text = L"Изменить пароль";

this->save\_pass\_button->UseVisualStyleBackColor = true;

this->save\_pass\_button->Click += gcnew System::EventHandler(this, &make\_new\_pass::Save\_pass\_button\_Click);

// label1

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 7.8F));

this->label1->Location = System::Drawing::Point(156, 52);

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

this->label1->Size = System::Drawing::Size(120, 17);

this->label1->TabIndex = 4;

this->label1->Text = L"Пароль изменён!";

this->label1->Visible = false;

// button1

this->button1->Location = System::Drawing::Point(146, 98);

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

this->button1->Size = System::Drawing::Size(140, 40);

this->button1->TabIndex = 5;

this->button1->Text = L"Хорошо";

this->button1->UseVisualStyleBackColor = true;

this->button1->Visible = false;

this->button1->Click += gcnew System::EventHandler(this, &make\_new\_pass::Button1\_Click);

// label2

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(142, 24);

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

this->label2->Size = System::Drawing::Size(163, 17);

this->label2->TabIndex = 6;

this->label2->Text = L"Введите новый пароль:";

this->label2->TextAlign = System::Drawing::ContentAlignment::TopCenter;

// maskedTextBox1

this->maskedTextBox1->Location = System::Drawing::Point(134, 70);

this->maskedTextBox1->Mask = L"aaaaaaaaaaaaaaaaaaaaa";

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

this->maskedTextBox1->PromptChar = ' ';

this->maskedTextBox1->Size = System::Drawing::Size(167, 22);

this->maskedTextBox1->TabIndex = 7;

this->maskedTextBox1->Visible = false;

this->maskedTextBox1->Click += gcnew System::EventHandler(this, &make\_new\_pass::MaskedTextBox1\_Click);

// make\_new\_pass

this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);

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

this->ClientSize = System::Drawing::Size(432, 178);

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

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

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

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

this->Controls->Add(this->save\_pass\_button);

this->Controls->Add(this->new\_pass\_textBox);

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

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

this->Name = L"make\_new\_pass";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Изменение пароля";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void Save\_pass\_button\_Click(System::Object^ sender, System::EventArgs^ e) {

//введённый пароль

if (is\_pass) {

cli::array<Char>^ pass\_buff;

pass\_buff = new\_pass\_textBox->Text->ToCharArray(); //преобразуем введённый пароль в массив символов

for (int i = 0; i < new\_pass\_textBox->TextLength; i++) { //шифруем пароль

pass\_buff[i] += 1;

pass\_input += (wchar\_t)pass\_buff[i];

}}

else {

pass\_input = score + "|" + maskedTextBox1->Text + "|" + DateTime::Now;

}

StreamWriter^ fps = gcnew StreamWriter(name\_file, true);

fps->Write(pass\_input);

fps->WriteLine();

delete fps;

new\_pass\_textBox->Hide();

save\_pass\_button->Hide();

label2->Hide();

button1->Show();

label1->Show();

maskedTextBox1->Hide(); }

private: System::Void Button1\_Click(System::Object^ sender, System::EventArgs^ e) {

this->Close(); }

private: System::Void MaskedTextBox1\_Click(System::Object^ sender, System::EventArgs^ e) {

maskedTextBox1->SelectionStart = maskedTextBox1->Text->Length; }

};

}

Password.h

#include "make\_new\_pass.h"

#pragma once

namespace airport {

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 password

/// </summary>

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

{

public:

int is\_admin = 0;

password(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~password()

{

if (components)

{

delete components;

}}

private: System::Windows::Forms::Button^ in\_button;

protected:

protected:

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

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(password::typeid));

this->in\_button = (gcnew System::Windows::Forms::Button());

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

this->SuspendLayout();

// in\_button

this->in\_button->Location = System::Drawing::Point(153, 111);

this->in\_button->Name = L"in\_button";

this->in\_button->Size = System::Drawing::Size(140, 40);

this->in\_button->TabIndex = 1;

this->in\_button->Text = L"Войти";

this->in\_button->UseVisualStyleBackColor = true;

this->in\_button->Click += gcnew System::EventHandler(this, &password::In\_button\_Click);

// textBox1

this->textBox1->Location = System::Drawing::Point(153, 58);

this->textBox1->MaxLength = 25;

this->textBox1->MinimumSize = System::Drawing::Size(140, 30);

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

this->textBox1->PasswordChar = '\*';

this->textBox1->Size = System::Drawing::Size(140, 30);

this->textBox1->TabIndex = 0;

this->textBox1->KeyPress += gcnew System::Windows::Forms::KeyPressEventHandler(this, &password::TextBox1\_KeyPress);

// password

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

this->ClientSize = System::Drawing::Size(432, 178);

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

this->Controls->Add(this->in\_button);

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

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

this->Name = L"password";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Авторизация";

this->Load += gcnew System::EventHandler(this, &password::Password\_Load);

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

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

cli::array<Char>^ pass\_buff;

String^ pass\_input; //введённый пароль

String^ pass\_right; //зашифрованный верный пароль

pass\_buff = textBox1->Text->ToCharArray(); //преобразуем введённый пароль в массив символов

for (int i = 0; i < textBox1->TextLength; i++) { //шифруем пароль

pass\_buff[i] += 1;

pass\_input += (wchar\_t)pass\_buff[i]; //преобразуем в system string }

StreamReader^ fps = gcnew StreamReader("password.txt");

pass\_right = fps->ReadLine();

fps->Close();

//сравнение паролей

if (pass\_right == pass\_input) {

is\_admin = 1;

this->Hide(); }

//сообщение о неверном пароле

else {

textBox1->Text = "";

MessageBox::Show("Неверный пароль!\nПовторите ввод.", "Ошибка",MessageBoxButtons::OK, MessageBoxIcon::Error);

}}

private: System::Void TextBox1\_KeyPress(System::Object^ sender, System::Windows::Forms::KeyPressEventArgs^ e) {

if (e->KeyChar == 13)

In\_button\_Click(sender, e); //активация клавиши по нажатию Enter

}

private: System::Void Password\_Load(System::Object^ sender, System::EventArgs^ e) {

}

}; };

Info.h

#pragma once

namespace airport {

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 info

/// </summary>

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

{

public:

info(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~info()

{

if (components)

{

delete components;

}}

private: System::Windows::Forms::RichTextBox^ richTextBox1;

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(info::typeid));

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->SuspendLayout();

// richTextBox1

this->richTextBox1->Location = System::Drawing::Point(0, 0);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->ReadOnly = true;

this->richTextBox1->Size = System::Drawing::Size(663, 477);

this->richTextBox1->TabIndex = 0;

this->richTextBox1->Text = resources->GetString(L"richTextBox1.Text");

// info

this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);

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

this->ClientSize = System::Drawing::Size(661, 465);

this->Controls->Add(this->richTextBox1);

this->ForeColor = System::Drawing::Color::Transparent;

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

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

this->Name = L"info";

this->Text = L"О программе";

this->ResumeLayout(false);

}

#pragma endregion };

}

Plus\_string.h

#pragma once

namespace airport {

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 plus\_string

/// </summary>

public ref class plus\_string : public System::Windows::Forms::Form

{

public:

DateTime now = DateTime::Now;

cli::array<DateTime>^ current\_times = gcnew array<DateTime>(5); //объявление массива дат

int count\_date = 0; //индекс для него

bool is\_bold = 0; //флаг: 1 - если дата уже выделена

int index\_bold;

String^ date\_str;

bool day\_correct = 0;

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

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

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

public:

String^ clear\_str = "";

int error\_month = 0; //флаг для ошибки месяца

plus\_string(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~plus\_string()

{

if (components)

{

delete components;

}}

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

public: System::Windows::Forms::MaskedTextBox^ num\_route\_Text;

public: System::Windows::Forms::MaskedTextBox^ type\_plane\_Text;

public: System::Windows::Forms::MaskedTextBox^ city\_Text;

public: System::Windows::Forms::MaskedTextBox^ time\_out\_Text;

public: System::Windows::Forms::MaskedTextBox^ time\_in\_Text;

public: System::Windows::Forms::MaskedTextBox^ price\_Text;

public: System::Windows::Forms::Button^ create\_button;

public: System::Windows::Forms::MonthCalendar^ monthCalendar1;

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

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(plus\_string::typeid));

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

this->num\_route\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->type\_plane\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->city\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->time\_out\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->time\_in\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->price\_Text = (gcnew System::Windows::Forms::MaskedTextBox());

this->create\_button = (gcnew System::Windows::Forms::Button());

this->monthCalendar1 = (gcnew System::Windows::Forms::MonthCalendar());

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->SuspendLayout();

// label1

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->label1->Location = System::Drawing::Point(35, 31);

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

this->label1->Size = System::Drawing::Size(163, 220);

this->label1->TabIndex = 0;

this->label1->Text = L"Номер рейса\r\n\r\nТип самолёта\r\n\r\nПункт назначения\r\n\r\nВремя вылета\r\n\r\nВремя прилёта\r"

L"\n\r\nЦена билета";

// num\_route\_Text

this->num\_route\_Text->AllowDrop = true;

this->num\_route\_Text->BeepOnError = true;

this->num\_route\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->num\_route\_Text->Location = System::Drawing::Point(180, 28);

this->num\_route\_Text->Mask = L"09999";

this->num\_route\_Text->Name = L"num\_route\_Text";

this->num\_route\_Text->PromptChar = ' ';

this->num\_route\_Text->Size = System::Drawing::Size(122, 27);

this->num\_route\_Text->TabIndex = 1;

this->num\_route\_Text->ValidatingType = System::Int32::typeid;

// type\_plane\_Text

this->type\_plane\_Text->BeepOnError = true;

this->type\_plane\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->type\_plane\_Text->Location = System::Drawing::Point(205, 67);

this->type\_plane\_Text->Mask = L"Aaaaa";

this->type\_plane\_Text->Name = L"type\_plane\_Text";

this->type\_plane\_Text->PromptChar = ' ';

this->type\_plane\_Text->Size = System::Drawing::Size(122, 27);

this->type\_plane\_Text->TabIndex = 2;

// city\_Text

this->city\_Text->BeepOnError = true;

this->city\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->city\_Text->Location = System::Drawing::Point(224, 106);

this->city\_Text->Mask = L"L\?\?\?\?\?\?\?\?\?";

this->city\_Text->Name = L"city\_Text";

this->city\_Text->PromptChar = ' ';

this->city\_Text->Size = System::Drawing::Size(122, 27);

this->city\_Text->TabIndex = 3;

// time\_out\_Text

this->time\_out\_Text->BeepOnError = true;

this->time\_out\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->time\_out\_Text->Location = System::Drawing::Point(224, 144);

this->time\_out\_Text->Mask = L"90:00";

this->time\_out\_Text->Name = L"time\_out\_Text";

this->time\_out\_Text->Size = System::Drawing::Size(122, 27);

this->time\_out\_Text->TabIndex = 5;

this->time\_out\_Text->ValidatingType = System::DateTime::typeid;

this->time\_out\_Text->Leave += gcnew System::EventHandler(this, &plus\_string::In\_time\_out\_maskedTextBox\_Leave);

// time\_in\_Text

this->time\_in\_Text->BeepOnError = true;

this->time\_in\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->time\_in\_Text->Location = System::Drawing::Point(205, 185);

this->time\_in\_Text->Mask = L"90:00";

this->time\_in\_Text->Name = L"time\_in\_Text";

this->time\_in\_Text->Size = System::Drawing::Size(122, 27);

this->time\_in\_Text->TabIndex = 6;

this->time\_in\_Text->ValidatingType = System::DateTime::typeid;

this->time\_in\_Text->Leave += gcnew System::EventHandler(this, &plus\_string::In\_time\_in\_maskedTextBox\_Leave);

// price\_Text

this->price\_Text->BeepOnError = true;

this->price\_Text->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->price\_Text->Location = System::Drawing::Point(180, 224);

this->price\_Text->Mask = L"9990.00";

this->price\_Text->Name = L"price\_Text";

this->price\_Text->Size = System::Drawing::Size(122, 27);

this->price\_Text->TabIndex = 7;

// create\_button

this->create\_button->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->create\_button->Location = System::Drawing::Point(252, 283);

this->create\_button->Name = L"create\_button";

this->create\_button->Size = System::Drawing::Size(116, 32);

this->create\_button->TabIndex = 8;

this->create\_button->Text = L"Создать";

this->create\_button->UseVisualStyleBackColor = true;

this->create\_button->Click += gcnew System::EventHandler(this, &plus\_string::Create\_button\_Click);

// monthCalendar1

this->monthCalendar1->Location = System::Drawing::Point(381, 44);

this->monthCalendar1->MaxDate = System::DateTime(2019, now.Month, 30, 0, 0, 0, 0);

this->monthCalendar1->MaxSelectionCount = 1;

this->monthCalendar1->MinDate = System::DateTime(2019, now.Month, 1, 0, 0, 0, 0);

this->monthCalendar1->Name = L"monthCalendar1";

this->monthCalendar1->TabIndex = 9;

this->monthCalendar1->DateSelected += gcnew System::Windows::Forms::DateRangeEventHandler(this, &plus\_string::MonthCalendar1\_DateSelected);

this->monthCalendar1->Leave += gcnew System::EventHandler(this, &plus\_string::MonthCalendar1\_Leave);

// label2

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->label2->Location = System::Drawing::Point(402, 9);

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

this->label2->Size = System::Drawing::Size(157, 20);

this->label2->TabIndex = 10;

this->label2->Text = L"Дни отправления";

// label3

this->label3->AutoSize = true;

this->label3->ForeColor = System::Drawing::Color::Red;

this->label3->Location = System::Drawing::Point(378, 251);

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

this->label3->Size = System::Drawing::Size(203, 17);

this->label3->TabIndex = 11;

this->label3->Text = L"Выберите хотя бы один день!";

this->label3->Visible = false;

// label4

this->label4->AutoSize = true;

this->label4->ForeColor = System::Drawing::Color::Red;

this->label4->Location = System::Drawing::Point(442, 298);

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

this->label4->Size = System::Drawing::Size(0, 17);

this->label4->TabIndex = 12;

// label5

this->label5->AutoSize = true;

this->label5->ForeColor = System::Drawing::Color::Red;

this->label5->Location = System::Drawing::Point(389, 268);

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

this->label5->Size = System::Drawing::Size(184, 34);

this->label5->TabIndex = 13;

this->label5->Text = L"Выберите дату в текущем \r\nмесяце!";

this->label5->TextAlign = System::Drawing::ContentAlignment::TopCenter;

this->label5->Visible = false;

// plus\_string

this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);

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

this->ClientSize = System::Drawing::Size(617, 327);

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

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

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

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

this->Controls->Add(this->monthCalendar1);

this->Controls->Add(this->create\_button);

this->Controls->Add(this->price\_Text);

this->Controls->Add(this->time\_in\_Text);

this->Controls->Add(this->time\_out\_Text);

this->Controls->Add(this->city\_Text);

this->Controls->Add(this->type\_plane\_Text);

this->Controls->Add(this->num\_route\_Text);

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

this->DoubleBuffered = true;

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

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

this->Name = L"plus\_string";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Добавить строку";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void In\_time\_out\_maskedTextBox\_Leave(System::Object^ sender, System::EventArgs^ e) {

cli::array <String^>^ pod\_string = time\_out\_Text->Text->Split(':');

if (time\_out\_Text->MaskCompleted) {

if (Convert::ToInt32(pod\_string[0]) > 23)

pod\_string[0] = "23";

if (Convert::ToInt32(pod\_string[1]) > 59)

pod\_string[1] = "59";

time\_out\_Text->Text = pod\_string[0] + pod\_string[1];

} }

private: System::Void In\_time\_in\_maskedTextBox\_Leave(System::Object^ sender, System::EventArgs^ e) {

cli::array <String^>^ pod\_string = time\_in\_Text->Text->Split(':');

if (time\_in\_Text->MaskCompleted) {

if (Convert::ToInt32(pod\_string[0]) > 23)

pod\_string[0] = "23";

if (Convert::ToInt32(pod\_string[1]) > 59)

pod\_string[1] = "59";

time\_in\_Text->Text = pod\_string[0] + pod\_string[1];

} }

private: System::Void MonthCalendar1\_Leave(System::Object^ sender, System::EventArgs^ e) {

input\_date(); }

int input\_date() {

if (!count\_date) {

label3->Show();

return 1;

}

else {

DateTime^ now\_day = gcnew DateTime();

int now\_month;

now\_day = DateTime::Now;

now\_month = now\_day->Month;

for (int i = 0; i < count\_date; i++) {

int cur\_month = current\_times[i].Month;

if (cur\_month != now\_month)

error\_month = 1; }

if (error\_month) {

label5->Show();

return 1; }

date\_str = "";

date\_str += current\_times[0].Day;

for (int i = 1; i < count\_date; i++)

{

date\_str += ", " + current\_times[i].Day; }

day\_correct = 1;

return 0;

} }

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

if (input\_date()) return;

if (!num\_route\_Text->MaskCompleted) {

MessageBox::Show("Некорректный номер рейса!", "Ошибка формата", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

if (!type\_plane\_Text->MaskCompleted) {

MessageBox::Show("Некорректный тип самолёта!", "Ошибка формата", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

if (!city\_Text->MaskCompleted) {

MessageBox::Show("Некорректный пункт назначения!", "Ошибка формата", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

if (!price\_Text->MaskCompleted) {

MessageBox::Show("Некорректная цена!", "Ошибка формата", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

if (!time\_in\_Text->MaskCompleted) {

MessageBox::Show("Некорректное время прилёта!", "Ошибка формата", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

if (!time\_out\_Text->MaskCompleted) {

MessageBox::Show("Некорректное время вылета!", "Ошибка формата", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

if (!count\_date) {

label3->Show();

return;

}

if (error\_month)

return;

clear\_str = num\_route\_Text->Text + "|" + type\_plane\_Text->Text + "|" + city\_Text->Text + "|" + date\_str + "|" + time\_out\_Text->Text + "|" + time\_in\_Text->Text + "|" + price\_Text->Text;

MessageBox::Show("Строка успешно добавлена!", "Успех", MessageBoxButtons::OK, MessageBoxIcon::Information);

this->Close(); }

private: System::Void MonthCalendar1\_DateSelected(System::Object^ sender, System::Windows::Forms::DateRangeEventArgs^ e) {

for (int i = 0; i < count\_date; i++)

if (DateTime::Equals(current\_times[i], monthCalendar1->SelectionStart)) {

is\_bold = 1;

index\_bold = i; }

if (is\_bold) {

monthCalendar1->RemoveBoldedDate(monthCalendar1->SelectionStart);

for (int i = index\_bold; i < count\_date - 1; i++)

current\_times[i] = current\_times[i + 1];

count\_date--;

is\_bold = 0;

}

else if (count\_date < 5)

{

current\_times[count\_date] = monthCalendar1->SelectionStart; //присваивает значение выделенной ячейки переменной

count\_date++;

monthCalendar1->AddBoldedDate(monthCalendar1->SelectionStart); //делает число жирным

}

else {

MessageBox::Show("Нельзя выбрать больше 5 дней отправления", "Ошибка ввода", MessageBoxButtons::OK, MessageBoxIcon::Hand);

}

monthCalendar1->UpdateBoldedDates(); //обновляет календарь

if (count\_date) label3->Hide();

label5->Hide();

error\_month = 0; } };

}

Game.h

#pragma once

#include "make\_new\_pass.h"

namespace airport {

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 game

/// </summary>

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

{

public:

int in\_cloud = 0;

float speed;

int score;

enum class dir\_plane { stay, right, left, down, up } where\_plane;

int is\_start = 0;

int size\_plane, widht\_cloud, height\_cloud;

private: System::Windows::Forms::Button^ game\_button;

private: System::Windows::Forms::Button^ rec\_button;

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

private: System::Windows::Forms::Label^ score\_text;

private: System::Windows::Forms::PictureBox^ pictureBox1;

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

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

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

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

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

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

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

public:

private: System::Windows::Forms::Button^ right\_button;

public:

void setup() {

speed = 2;

timer1->Start();

score\_text->Text = Convert::ToString(1);

size\_plane = 100;

height\_cloud = 75;

widht\_cloud = 120;

loc\_x = 320;

loc\_y = 400;

where\_plane = dir\_plane::stay;

cloud\_xy\_coord[0, 1] = -100;

cloud\_xy\_coord[0, 0] = 20;

cloud\_xy\_coord[1, 1] = -100;

cloud\_xy\_coord[1, 0] = 220;

cloud\_xy\_coord[2, 1] = -100;

cloud\_xy\_coord[2, 0] = 520;

cloud\_xy\_coord[3, 1] = -330;

cloud\_xy\_coord[3, 0] = 120;

cloud\_xy\_coord[4, 1] = -330;

cloud\_xy\_coord[4, 0] = 270;

cloud\_xy\_coord[5, 1] = -330;

cloud\_xy\_coord[5, 0] = 320;

cloud\_xy\_coord[6, 1] = -580;

cloud\_xy\_coord[6, 0] = 20;

cloud\_xy\_coord[7, 1] = -580;

cloud\_xy\_coord[7, 0] = 120;

cloud\_xy\_coord[8, 1] = -580;

cloud\_xy\_coord[8, 0] = 620;

};

array<int, 2> ^ cloud\_xy\_coord = gcnew array<int, 2>(9, 2);

int loc\_x;

int loc\_y;

void promotion(int x, int y) {

if ((loc\_x == 0 && x > 0) || (loc\_x == (this->Width - 100) && x < 0) || (loc\_x > 0 && loc\_x < (this->Width - 100)))

loc\_x += x;

if ((loc\_y == 400 && y < 0) || (loc\_y == 0 && y > 0) || (loc\_y > 0 && loc\_y < 400))

loc\_y += y;

}

Bitmap^ cloud = gcnew Bitmap("cloud.png");

private: System::Windows::Forms::Timer^ timer1;

public:

Bitmap^ plane = gcnew Bitmap("plane.png");

game(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

SetStyle(ControlStyles::OptimizedDoubleBuffer | ControlStyles::AllPaintingInWmPaint | ControlStyles::UserPaint, true);

UpdateStyles();

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~game()

{

if (components)

{

delete components;

}}

private: System::ComponentModel::IContainer^ components;

protected:

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(game::typeid));

System::Windows::Forms::DataGridViewCellStyle^ dataGridViewCellStyle1 = (gcnew System::Windows::Forms::DataGridViewCellStyle());

System::Windows::Forms::DataGridViewCellStyle^ dataGridViewCellStyle2 = (gcnew System::Windows::Forms::DataGridViewCellStyle());

this->timer1 = (gcnew System::Windows::Forms::Timer(this->components));

this->game\_button = (gcnew System::Windows::Forms::Button());

this->rec\_button = (gcnew System::Windows::Forms::Button());

this->right\_button = (gcnew System::Windows::Forms::Button());

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

this->score\_text = (gcnew System::Windows::Forms::Label());

this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());

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

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

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

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

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

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

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

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->BeginInit();

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

this->SuspendLayout();

// timer1

this->timer1->Interval = 25;

this->timer1->Tick += gcnew System::EventHandler(this, &game::Timer1\_Tick);

// game\_button

this->game\_button->BackColor = System::Drawing::Color::Azure;

this->game\_button->FlatAppearance->BorderColor = System::Drawing::Color::White;

this->game\_button->FlatAppearance->BorderSize = 0;

this->game\_button->Font = (gcnew System::Drawing::Font(L"Sylfaen", 16.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

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

this->game\_button->ForeColor = System::Drawing::Color::DarkBlue;

this->game\_button->Location = System::Drawing::Point(275, 139);

this->game\_button->Name = L"game\_button";

this->game\_button->Size = System::Drawing::Size(190, 64);

this->game\_button->TabIndex = 0;

this->game\_button->Text = L"Играть";

this->game\_button->UseVisualStyleBackColor = false;

this->game\_button->Click += gcnew System::EventHandler(this, &game::Game\_button\_Click);

// rec\_button

this->rec\_button->BackColor = System::Drawing::Color::Azure;

this->rec\_button->FlatAppearance->BorderColor = System::Drawing::Color::White;

this->rec\_button->FlatAppearance->BorderSize = 0;

this->rec\_button->Font = (gcnew System::Drawing::Font(L"Sylfaen", 16.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

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

this->rec\_button->ForeColor = System::Drawing::Color::DarkBlue;

this->rec\_button->Location = System::Drawing::Point(275, 316);

this->rec\_button->Name = L"rec\_button";

this->rec\_button->Size = System::Drawing::Size(190, 64);

this->rec\_button->TabIndex = 1;

this->rec\_button->Text = L"Рекорды";

this->rec\_button->UseVisualStyleBackColor = false;

this->rec\_button->Click += gcnew System::EventHandler(this, &game::Rec\_button\_Click);

// right\_button

this->right\_button->BackColor = System::Drawing::Color::Azure;

this->right\_button->FlatAppearance->BorderColor = System::Drawing::Color::White;

this->right\_button->FlatAppearance->BorderSize = 0;

this->right\_button->Font = (gcnew System::Drawing::Font(L"Sylfaen", 16.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

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

this->right\_button->ForeColor = System::Drawing::Color::DarkBlue;

this->right\_button->Location = System::Drawing::Point(275, 227);

this->right\_button->Name = L"right\_button";

this->right\_button->Size = System::Drawing::Size(190, 64);

this->right\_button->TabIndex = 2;

this->right\_button->Text = L"Правила";

this->right\_button->UseVisualStyleBackColor = false;

this->right\_button->Click += gcnew System::EventHandler(this, &game::Right\_button\_Click);

// label1

this->label1->AutoSize = true;

this->label1->BackColor = System::Drawing::Color::Transparent;

this->label1->Font = (gcnew System::Drawing::Font(L"Sylfaen", 24, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

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

this->label1->ForeColor = System::Drawing::Color::Navy;

this->label1->Location = System::Drawing::Point(112, 32);

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

this->label1->Size = System::Drawing::Size(532, 104);

this->label1->TabIndex = 3;

this->label1->Text = L"Пилотируйте свой самолёт \r\n\"To Dublin\"!";

this->label1->TextAlign = System::Drawing::ContentAlignment::MiddleCenter;

// score\_text

this->score\_text->AutoSize = true;

this->score\_text->BackColor = System::Drawing::Color::Transparent;

this->score\_text->Font = (gcnew System::Drawing::Font(L"Sylfaen", 10.8F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

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

this->score\_text->ForeColor = System::Drawing::Color::Navy;

this->score\_text->Location = System::Drawing::Point(22, 32);

this->score\_text->Name = L"score\_text";

this->score\_text->Size = System::Drawing::Size(23, 25);

this->score\_text->TabIndex = 4;

this->score\_text->Text = L"0";

this->score\_text->Visible = false;

// pictureBox1

this->pictureBox1->BackColor = System::Drawing::Color::Transparent;

this->pictureBox1->Image = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"pictureBox1.Image")));

this->pictureBox1->Location = System::Drawing::Point(307, 407);

this->pictureBox1->Name = L"pictureBox1";

this->pictureBox1->Size = System::Drawing::Size(130, 134);

this->pictureBox1->TabIndex = 5;

this->pictureBox1->TabStop = false;

// dataGridView1

this->dataGridView1->AllowUserToAddRows = false;

this->dataGridView1->AllowUserToDeleteRows = false;

this->dataGridView1->AllowUserToResizeRows = false;

this->dataGridView1->AutoSizeColumnsMode = System::Windows::Forms::DataGridViewAutoSizeColumnsMode::Fill;

this->dataGridView1->BackgroundColor = System::Drawing::Color::LightSkyBlue;

this->dataGridView1->BorderStyle = System::Windows::Forms::BorderStyle::None;

this->dataGridView1->CellBorderStyle = System::Windows::Forms::DataGridViewCellBorderStyle::Raised;

dataGridViewCellStyle1->Alignment = System::Windows::Forms::DataGridViewContentAlignment::TopCenter;

dataGridViewCellStyle1->BackColor = System::Drawing::Color::White;

dataGridViewCellStyle1->Font = (gcnew System::Drawing::Font(L"Sylfaen", 10, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

dataGridViewCellStyle1->ForeColor = System::Drawing::SystemColors::WindowText;

dataGridViewCellStyle1->SelectionBackColor = System::Drawing::SystemColors::Highlight;

dataGridViewCellStyle1->SelectionForeColor = System::Drawing::SystemColors::HighlightText;

dataGridViewCellStyle1->WrapMode = System::Windows::Forms::DataGridViewTriState::True;

this->dataGridView1->ColumnHeadersDefaultCellStyle = dataGridViewCellStyle1;

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

this->dataGridView1->Columns->AddRange(gcnew cli::array< System::Windows::Forms::DataGridViewColumn^ >(3) {

this->count,

this->name, this->time });

this->dataGridView1->GridColor = System::Drawing::Color::Navy;

this->dataGridView1->Location = System::Drawing::Point(121, 139);

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

dataGridViewCellStyle2->Alignment = System::Windows::Forms::DataGridViewContentAlignment::TopLeft;

dataGridViewCellStyle2->BackColor = System::Drawing::SystemColors::Control;

dataGridViewCellStyle2->Font = (gcnew System::Drawing::Font(L"Sylfaen", 10, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

dataGridViewCellStyle2->ForeColor = System::Drawing::SystemColors::WindowText;

dataGridViewCellStyle2->SelectionBackColor = System::Drawing::SystemColors::Highlight;

dataGridViewCellStyle2->SelectionForeColor = System::Drawing::SystemColors::HighlightText;

dataGridViewCellStyle2->WrapMode = System::Windows::Forms::DataGridViewTriState::True;

this->dataGridView1->RowHeadersDefaultCellStyle = dataGridViewCellStyle2;

this->dataGridView1->RowHeadersVisible = false;

this->dataGridView1->RowHeadersWidth = 51;

this->dataGridView1->RowTemplate->Height = 24;

this->dataGridView1->RowTemplate->ReadOnly = true;

this->dataGridView1->ScrollBars = System::Windows::Forms::ScrollBars::Vertical;

this->dataGridView1->Size = System::Drawing::Size(497, 262);

this->dataGridView1->TabIndex = 6;

this->dataGridView1->Visible = false;

// count

this->count->FillWeight = 44.73458F;

this->count->HeaderText = L"Счет";

this->count->MinimumWidth = 6;

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

// name

this->name->FillWeight = 80.21391F;

this->name->HeaderText = L"Игрок";

this->name->MinimumWidth = 6;

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

// time

this->time->FillWeight = 175.0515F;

this->time->HeaderText = L"Дата и время";

this->time->MinimumWidth = 6;

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

// button1

this->button1->Location = System::Drawing::Point(546, 111);

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

this->button1->Size = System::Drawing::Size(75, 29);

this->button1->TabIndex = 7;

this->button1->Text = L"Назад";

this->button1->UseVisualStyleBackColor = true;

this->button1->Visible = false;

this->button1->Click += gcnew System::EventHandler(this, &game::Button1\_Click);

// label2

this->label2->AutoSize = true;

this->label2->BackColor = System::Drawing::Color::Transparent;

this->label2->ForeColor = System::Drawing::Color::Navy;

this->label2->Location = System::Drawing::Point(119, 131);

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

this->label2->Size = System::Drawing::Size(502, 286);

this->label2->TabIndex = 8;

this->label2->Text = resources->GetString(L"label2.Text");

this->label2->TextAlign = System::Drawing::ContentAlignment::TopCenter;

this->label2->Visible = false;

// label3

this->label3->AutoSize = true;

this->label3->BackColor = System::Drawing::Color::Transparent;

this->label3->Font = (gcnew System::Drawing::Font(L"Sylfaen", 10.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

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

this->label3->ForeColor = System::Drawing::Color::Red;

this->label3->Location = System::Drawing::Point(502, 13);

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

this->label3->Size = System::Drawing::Size(53, 22);

this->label3->TabIndex = 9;

this->label3->Text = L"-300!";

this->label3->Visible = false;

// game

this->AutoScaleDimensions = System::Drawing::SizeF(10, 22);

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

this->BackColor = System::Drawing::Color::LightSkyBlue;

this->ClientSize = System::Drawing::Size(732, 553);

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

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

this->Controls->Add(this->score\_text);

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

this->Controls->Add(this->right\_button);

this->Controls->Add(this->rec\_button);

this->Controls->Add(this->game\_button);

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

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

this->Controls->Add(this->pictureBox1);

this->Font = (gcnew System::Drawing::Font(L"Sylfaen", 10, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

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

this->KeyPreview = true;

this->Name = L"game";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"To Dublin";

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

this->Paint += gcnew System::Windows::Forms::PaintEventHandler(this, &game::Game\_Paint);

this->KeyDown += gcnew System::Windows::Forms::KeyEventHandler(this, &game::Game\_KeyDown);

this->KeyUp += gcnew System::Windows::Forms::KeyEventHandler(this, &game::Game\_KeyUp);

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->EndInit();

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

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void Game\_Paint(System::Object^ sender, System::Windows::Forms::PaintEventArgs^ e) {

promotion(0, 0);

Graphics^ g = e->Graphics;

//отрисовка самолёта

g->DrawImage(plane, Rectangle(loc\_x, loc\_y, size\_plane, size\_plane));

cli::array <Graphics^>^ g\_cloud;

//отрисовка облака

for (int i = 0; i < 9; i++)

{

g->DrawImage(cloud, Rectangle(cloud\_xy\_coord[i, 0], cloud\_xy\_coord[i, 1], widht\_cloud, height\_cloud));

} }

private: System::Void Timer1\_Tick(System::Object^ sender, System::EventArgs^ e) {

score\_text->Text = Convert::ToString(Convert::ToInt32(score\_text->Text) + 1);

Refresh();

move\_object();

if (Convert::ToInt32(score\_text->Text) % 300 == 0)

speed+= 0.5; }

private: System::Void Game\_KeyDown(System::Object^ sender, System::Windows::Forms::KeyEventArgs^ e) {

switch (e->KeyCode) {

case Keys::Right: where\_plane = dir\_plane::right; break;

case Keys::Left: where\_plane = dir\_plane::left; break;

case Keys::Down: where\_plane = dir\_plane::down; break;

case Keys::Up: where\_plane = dir\_plane::up; break;

} }

void move\_object() {

Random rnd;

for (int i = 0; i < 9; i++) {

//проверка на встречу крыла самолёта с облаком по OY

if ((loc\_y + size\_plane \* 0.4 - cloud\_xy\_coord[i, 1]) > 0 && (loc\_y + size\_plane \* 0.6 - ((cloud\_xy\_coord[i, 1]) + height\_cloud)) < 0) {

//слева

if (!in\_cloud && (loc\_x - cloud\_xy\_coord[i, 0]) > 0 && (loc\_x - (cloud\_xy\_coord[i, 0] + widht\_cloud)) < 0) {

label3->Location = Point(loc\_x - 50, loc\_y);

label3->Visible = true;

score\_text->Text = Convert::ToString(Convert::ToInt32(score\_text->Text) - 300);

in\_cloud = i + 1;

}

//српва

else if (!in\_cloud && (cloud\_xy\_coord[i, 0] - loc\_x) < size\_plane && (cloud\_xy\_coord[i, 0] - loc\_x) > 0) {

label3->Location = Point(loc\_x + 150, loc\_y);

label3->Visible = true;

score\_text->Text = Convert::ToString(Convert::ToInt32(score\_text->Text) - 300);

in\_cloud = i + 1;

}

}

else if (i + 1 == in\_cloud) {

in\_cloud = 0;

label3->Visible = false;

}

//проверка на встречу облака и самолёта

if ((loc\_x + size\_plane \* 0.7 - cloud\_xy\_coord[i, 0]) > 0 && (loc\_x + size\_plane \* 0.3 - (cloud\_xy\_coord[i, 0] + widht\_cloud)) < 0 && (loc\_y - cloud\_xy\_coord[i, 1]) > size\_plane \* -0.5 && (loc\_y + size\_plane \* 0.9 - ((cloud\_xy\_coord[i, 1]) + height\_cloud)) < size\_plane \* 0.5) {

is\_start = false;

score = Convert::ToInt32(score\_text->Text)/10;

timer1->Stop();

where\_plane = dir\_plane::stay;

make\_new\_pass^ new\_rec = gcnew make\_new\_pass();

//изменяем свойства вызываемой формы для записи рекордов

new\_rec->is\_pass = false;

new\_rec->name\_file = "record.txt";

new\_rec->label2->Text = "Вы набрали " + score + " очков!\nВведите своё имя:";

new\_rec->label2->Font = (gcnew System::Drawing::Font(L"Sylfaen", 10, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

new\_rec->Text = "Добавление рекорда";

new\_rec->save\_pass\_button->Text = "Добавить";

new\_rec->Font = (gcnew System::Drawing::Font(L"Sylfaen", 10, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

new\_rec->save\_pass\_button->Font = (gcnew System::Drawing::Font(L"Sylfaen", 10, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

new\_rec->save\_pass\_button->Show();

new\_rec->label1->Text = "Результат добавлен!";

new\_rec->score = score;

new\_rec->maskedTextBox1->Visible = true;

new\_rec->new\_pass\_textBox->Visible = false;

new\_rec->Show();

new\_rec->Owner = this;

game\_button->Show();

right\_button->Show();

rec\_button->Show();

label1->Show();

score\_text->Hide();

}

if (cloud\_xy\_coord[i, 1] > 600) { //проверка на выход облака за границы

cloud\_xy\_coord[i, 1] = -100; //возвращаем у-координату облака в начало

cloud\_xy\_coord[i, 0] = rnd.Next(650); //задаем рандомную х-координату облака

}

}

for (int i = 0; i < 9; i++)

cloud\_xy\_coord[i, 1] += speed;

int x = 5;

switch (where\_plane) {

case dir\_plane::right: promotion(x, 0); break;

case dir\_plane::left: promotion(-x, 0); break;

case dir\_plane::down: promotion(0, x); break;

case dir\_plane::up: promotion(0, -x); break;

}

}

private: System::Void Game\_KeyUp(System::Object^ sender, System::Windows::Forms::KeyEventArgs^ e) {

where\_plane = dir\_plane::stay; }

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

is\_start = true;

if (is\_start) {

setup();

game\_button->Hide();

right\_button->Hide();

rec\_button->Hide();

label1->Hide();

score\_text->Show();

pictureBox1->Hide();

} }

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

game\_button->Hide();

rec\_button->Hide();

right\_button->Hide();

button1->Show();

label2->Show(); }

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

game\_button->Hide();

rec\_button->Hide();

right\_button->Hide();

button1->Visible = true;

dataGridView1->Visible = true;

if (!IO::File::Exists("record.txt")) {

StreamWriter^ fps = gcnew StreamWriter("record.txt", true);

fps->Close();

label2->Text = "\n\n\nВ таблице пока нет рекордов!";

label2->Visible = true;

}

StreamReader^ str = gcnew StreamReader("record.txt");

String^ buf\_string;

dataGridView1->Rows->Clear();

for (int i = 0; !str->EndOfStream; i++) {

dataGridView1->Rows->Add();

buf\_string = str->ReadLine();

cli::array<String^>^ cells\_fill;

cells\_fill = buf\_string->Split('|');

for (int j = 0; j < 3; j++) {

dataGridView1->Rows[i]->Cells[j]->Value = cells\_fill[j];

}}

str->Close(); }

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

timer1->Stop(); }

private: System::Void Button1\_Click(System::Object^ sender, System::EventArgs^ e) {

dataGridView1->Visible = false;

button1->Visible = false;

label2->Visible = false;

game\_button->Show();

rec\_button->Show();

right\_button->Show();}};}

# Приложение В Руководство системного администратора

На переносном диске содержится архивированная папка с программой и её исходным кодом. Для работы с данной программой необходимо разархивировать данную папку и запустить airport.exe, находящийся по адресу \toys\Release

Для полноценного функционирования программы требуется:

* ОС Windows XP SP3 или выше;
* Оперативная память 10МБ или выше;
* Жесткий диск: 100МБ для хранения программы; дополнительное место для хранения вводимых в базу данных;
* .NET Framework версии 2.0 или выше;
* Наличие мыши (или её аналоги), клавиатуры, монитора;
* Разрешение экрана 1920 x 1080.

# Приложение Г Руководство пользователя

Для запуска программы требуется дважды щелкнуть по файлу airport.exe.

В открывшемся окне выбирите кнопку «Администратор» или «Пользователь», в зависимости от доступных вам прав доступа. Кнопка «Выход» закрывает программу, а «Справка» вызывает окно с информацией об основных компонентах и возможностях программы

В интерфейсе пользователя вам доступны следующие пункты горизонтального меню сверху:

**1.Справка**

Представляет собой окно с информацией об основных компонентах и возможностях программы

1. **Запрос**

При вызове данной команды справа в форме появятся поля для обработки интересующего запроса. Просто введите необходимые данные или сотрите их при, для того чтобы ввести новый запрос. Если данных слишком много, можно закрыть запрос, нажав на пункт «Запрос» в меню или черный крестик внизу запроса.

1. **Играть**

Здесь вы можете сыграть в игру «To Dublin».

* + 1. Играть – запуск игры
    2. Рекорды – транслирует таблицу рекордов
    3. Правила игры – показывает правила игры

Правила игры: цель игры – провести свой самолёт как можно ближе к пункту назначения, не врезаясь в облака. Осторожннее, за это могут снять баллы.

Будьте внимательны! Скорость самолёта, а, значит, и сложность тоже, увеличивается со временем.

После окончания игры, приложение предложит вам вснести свои данные в таблицу рекордов.

Управление самолётом осуществляется клавишами вверх, вниз, влево, вправо.

1. **Выход из БД**

Аналогичен крестику в верхнем правом углу. Возвращает вас в форму авторизации.

В режиме пользователя вам будут доступны не только все вышеперечисленные, но и дополнительные пункты меню. А именно:

1. **Редактирование**, с подпунктами
   1. **Редактировать запись**

Справа появится дополнительный столбец с ячейками для выбора строк. Выберите строки, которые вы хотите редактировать и нажмите на значок редактирования, появившийся в правом нижнем углу. Для каждой из выбраннх вами строк появиться форма редактирования строки, причем старые данные не будут утеряны.

Внесите вне необходимые изменения, соблюдая следующие форматы запроса.

* + Номер рейса – целочисленное число;
  + Тип самолёта – строка, состоящая из букв и цифр;
  + Пункт назначения – наименование города – строка, состоящая из букв;
  + Дни отправления – строа из нескольких целочисленных элементов, разделённых запятыми;
  + Время вылета – время в формате 23:59;
  + Время прилёта – время в формате 23:59;
  + Цена билета – число в формате с плавающей точкой.

В случае неверного формата, будет выведено сообения об ошибке. Изменения нельзя сохранить, пока не будут исправлены все ошибки.

* 1. **Добавить запись**

В новом окне появится форма для добавления записи, аналогичная форме для редактирования.

* 1. **Удалить запись**

Справа появится дополнительный столбец с ячейками для выбора строк. Выберите строки, которые вы хотите удалить и нажмите на значок удаления, появившийся в правом нижнем углу. Каждая из выбранных вами строк будет удалена. Будьде осторожны! Утерянную информацию восстановить невозможно.

1. **Работа с БД, с подпунктами:**
   1. Сохранить изменения

Сохраняет все внесённые вами изменения в документе.

* 1. Изменить пароль

Вызовёт окно для изменения пароля. В появившемся окне введите пароль и нажмите кнопку «Изменить пароль». Проследите за тем, чтобы никто не видел ваш пароль!

В случае утери пароля, для восстановления доступа к учетной записи «Администратор» обратитесь в техподдержку фирмы, выпустившей это ПО.

1. **Иконка сохранения** (в форме дискеты) окрашена в черный цвет, когда данные сохранены и в красный, когда в программе существуют несохранённые изменения.