Министерство образования и науки РФ

Государственное образовательное учреждение

Высшего образования

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (ВлГУ)

Кафедра «Физика и прикладная математика» (ВлГУ)

Курсовая работа

По дисциплине «Прикладное программирование»

«Программный модуль авиакасса.»

Выполнил:

Ст. ПКсп-114

Приняла: Ст. преподаватель  
 Шишкина М. В.

Владимир 2017 г.

**Содержание.**

[**Введение. 3**](#_Toc477129285)

[**Глава1. Постановка задачи 4**](#_Toc477129286)

[**Глава 2. Выбор системы программирования 5**](#_Toc477129287)

[**Глава 3. Описание процесса проектирования приложения 6**](#_Toc477129288)

[**3.1.Взаимосвязь программных модулей 6**](#_Toc477129289)

[**3.1.1.Функция подборки рейса. 6**](#_Toc477129290)

[**3.1.2.Функция добавления рейса. 7**](#_Toc477129291)

[**3.1.3.Функция покупки билета. 8**](#_Toc477129292)

[**3.1.4. Функция очистки полей на вкладке “Настройка расписания” 9**](#_Toc477129293)

[**3.1.5.Функции удаление рейса из lvRaspisanie. 9**](#_Toc477129294)

[**3.1.5.1 Удаление первой строки. 9**](#_Toc477129295)

[**3.1.5.2 Удаление строки под номером. 10**](#_Toc477129296)

[**3.1.5.3 Удаление всех строк. 10**](#_Toc477129297)

[**3.1.6. Функция очистки полей информации о пассажире. 11**](#_Toc477129298)

[**3.2. Схема работы программы. 11**](#_Toc477129299)

[**3.3. Проектирование интерфейса. 12**](#_Toc477129300)

[**Глава 4. Инструкция для пользователя по работе с приложением 15**](#_Toc477129301)

[**Заключение. 16**](#_Toc477129302)

[**Список использованных информационных источников: 17**](#_Toc477129303)

[**Приложение 1 18**](#_Toc477129304)

# **Введение.**

Компьютеры уже давно и прочно вошли в нашу жизнь. Они помогают решать задачи, на которые раньше приходилось тратить большие объемы времени и человеческих сил. Они развлекают и работают за нас. Именно с выполнением работы компьютером за человека связана моя курсовая работа.

Одна из самых утомляющих работ – ручное заполнение тысяч одинаковых форм, содержащих, к примеру, информацию о авиаперелетах. Во всех крупных компаниях уже давно перешли на электронную авиакассу.

В данной работе я рассмотрю все этапы создания программного модуля “Авиакасса”.

# **Глава1. Постановка задачи**

В теме моей курсовой работы требуется создать приложение “Авиакасса”, которое бы использовалось в аэропорту и облегчало работу кассиру. В качестве входных данных используется интересующий покупателя рейс, программа подбирает подходящие рейсы по данному критерию, далее оформляется покупка билета. Покупатель на выходе получает билет на рейс, в виде текстового файла.

# **Глава 2. Выбор системы программирования**

Для выполнения поставленной задачи, в качестве системы программирования, я выбрал C#. Потому что, С# полностью обьектно- ориентирован и поддерживает абсолютно все классы и пространства имен платформы .NET Framework. С# отлично подходит для прикладного программирования под платформу Windows.

# **Глава 3. Описание процесса проектирования приложения**

# **3.1.Взаимосвязь программных модулей**

Итак, первым делом необходимо определиться с функционалом, которым будет обладать модуль. Во-первых, мы должны иметь возможность добавлять рейс в базу данных. Во-вторых, мы должны возможность искать интересующий покупателя рейс. В-третьих, должна быть возможность удалять рейс из базы данных, чтобы внести в него какие-либо изменения или перенести его. И в-четвертых, должна реализовываться регистрация пассажира. Информация о пассажире должна содержать: Ф.И.О, серия и номер документа (паспорта или заграничного паспорта), пол, дату рождения, гражданство и срок действия предоставляемого документа. Информация о рейсе должна содержать: город отправления, город прибытия, дата и время отправления, дата и время прибытия, вид самолета, класс билета, цена перелета.

Поиск должен осуществляться по любому из полей информации о рейсе.

Форма была разделена на 3 вкладки: 1- Настройка расписания. 2- Расписание самолетов. 3- Покупка билета.

# **3.1.1.Функция подборки рейса.**

private void btnSeachReis\_Click\_1(object sender, EventArgs e)

{

lbCheckReis.Items.Clear();

if (cbSeachOut.Text == string.Empty)

return;

if (cbSeachIn.Text == string.Empty)

return;

if (cbSeachKlass.Text == string.Empty)

return;

string date = dtbDateOut.Value.Date.ToShortDateString();

foreach (ListViewItem item in this.lvRaspisanie.Items)

{

if ((item.SubItems[0].Text == cbSeachOut.Text) && (item.SubItems[1].Text == cbSeachIn.Text) && (item.SubItems[5].Text == cbSeachKlass.Text) && (item.SubItems[2].Text.Remove(10) == date))

{

item.ForeColor = Color.Red;

lbCheckReis.Items.Add(item.SubItems[0].Text + " " + item.SubItems[1].Text + " " + item.SubItems[2].Text + " " + item.SubItems[3].Text + " " + item.SubItems[4].Text + " " + item.SubItems[5].Text + " " + item.SubItems[6].Text);

}

else

item.ForeColor = Color.Black;

}

}

После нажатия кнопки управление передается обработчику события “btnSeachReis\_Click\_1”. Сперва программа проверяет, все ли поля заполнены. После создается переменная строкового типа “date” – в которую с помощь метода “ToShortDateString()” записываются значения с dtbDateOut. Далее программа ищет совпадения данных введенных пользователем с элементами в ListView, при совпадении перекрашивает элемент в красный цвет и записывает данные хранящиеся в этом элементе, в lbCheckReis. Если данные не совпадают с элементом в ListView, программа окрашивает элемент в черный цвет.

# **3.1.2.Функция добавления рейса.**

private void btnAdd\_Click\_1(object sender, EventArgs e)

{

ListViewItem ListItem = new ListViewItem(cbPlusOut.Text);

ListItem.SubItems.Add(cbPlusIn.Text);

ListItem.SubItems.Add(mtbDateOut.Text);

ListItem.SubItems.Add(mtbDateIn.Text);

ListItem.SubItems.Add(cbTypeAir.Text);

ListItem.SubItems.Add(cbKlass.Text);

ListItem.SubItems.Add(mtbPrice.Text);

lvRaspisanie.Items.Add(ListItem);

MessageBox.Show("Данные добавлены!", "Авиакасс");

saveToolStripMenu();

}

После нажатия кнопки управление передается обработчику события btnAdd\_Click\_1. Далее инициализируется новый объект ListViewItem, выдается сообщение, в отдельном окне, о том, что Данные добавлены!, далее программа переходит в функцию записи данных в файл.

private void saveToolStripMenu()//записывает данные в файл.

{

isWrite = true;

StringBuilder listViewContent = new StringBuilder();

foreach (ListViewItem item in this.lvRraspisanie.Items)

{

foreach (ListViewItem.ListViewSubItem subItem in item.SubItems)

{

listViewContent.Append(subItem.Text); listViewContent.Append('|');

}

listViewContent.Append(Environment.NewLine);

}

using (TextWriter tw = new StreamWriter(UsersDirectory + "\\airkassa.txt"))

{

tw.WriteLine(listViewContent.ToString());

tw.Close();

}

}

# **3.1.3.Функция покупки билета.**

private void lbCheckReis\_MouseDoubleClick\_1(object sender, MouseEventArgs e)

{

StreamWriter ticket = File.CreateText(UsersDirectory + "\\ticket.txt");

ticket.WriteLine(" Билет ");

ticket.WriteLine(" Данные о пассажире ");

ticket.WriteLine("Фамилия - " + tbFamily.Text);

ticket.WriteLine("Имя - " + tbName.Text);

ticket.WriteLine("Дата рождения - " + dtpBithday.Value.Date.ToShortDateString());

ticket.WriteLine("Гражданство - " + tbGrazh.Text);

ticket.WriteLine("Серия и № документа - " + tbNumDoc.Text);

ticket.WriteLine("Срок действия документа - " + dtpSrok.Value.Date.ToShortDateString());

ticket.WriteLine(" Информация о полете");

ticket.WriteLine(lbcheckReis.SelectedItem);

ticket.Flush();

ticket.Close();

}

При двойном клике на интересующих элемент в lbCheckReis, управление передается обработчику события lbCheckReis\_MouseDoubleClick\_1. Программа создает в директории запуска программы текстовый файл “ticket.txt” и записывает в него информацию о билете, которая берется из соответствующих элементов на форме. После очищается буфер и файл закрывается. Далее требуется нажать кнопку “Покупка билета”, управление передастся обработчику события btnBuyTic\_Click\_1.

private void btnBuyTic\_Click\_1(object sender, EventArgs e)

{

rtbTiccet.Text = "";

StreamReader ticket = File.OpenText(UsersDirectory + "\\ticket.txt");

while (!ticket.EndOfStream)

rtb\_Ticcet.Text += ticket.ReadLine() + "\n";

ticket.Close();

}

rtbTiccet очищается, открывается файл ticket.txt. Пока свойство EndOfStream выдает значение false, на rtbTiccet выводится строчка из файла, когда свойство EndOfStream получает значение true, файл закрывается.

# **3.1.4. Функция очистки полей на вкладке “Настройка расписания”**

private void btnClearReis\_Click\_1(object sender, EventArgs e)

{

cbPlusOut.Text = "Откуда";

cbPlusIn.Text = "Куда";

cbTypeAir.Text = "Тип самолета";

cbKlass.Text = "Класс билета";

mtbDateOut.Clear();

mtbDateIn.Clear();

mtbPrice.Clear();

MessageBox.Show("Поля очищены!", "Авиакасса");

}

После нажатия кнопки управление передается обработчику события btnClearReis\_Click\_1. Программа предается всем элементам значения, заданные по умолчанию и выдается в отдельном окне сообщение о том, что поля очищены.

# **3.1.5.Функции удаление рейса из lvRaspisanie.**

# **3.1.5.1 Удаление первой строки.**

private void btnClearStr1\_Click\_1(object sender, EventArgs e)

{

if (lvRaspisanie.Items.Count > 0)

lvRaspisanie.Items.Remove(lvRaspisanie.Items[0]);

MessageBox.Show("Первая строка удалена! ", "Авиакасса");

saveToolStripMenu();

}

После нажатия кнопки управление передается обработчику события btnClearStr1 Программа проверяет есть ли элементы в lvRaspisanie, если они есть удаляет нулевой элемент, далее выдает в отдельном окне сообщение о том, что первая строка удалена, и переходит в функцию записи данных в файл

# **3.1.5.2 Удаление строки под номером.**

private void btnClearStrNum\_Click\_1(object sender, EventArgs e)

{

string m = tbNumStr.Text;

int b = Convert.ToInt32(m);

int r = b - 1;

if (lvRaspisanie.Items.Count > 0)

lvRaspisanie.Items.Remove(lv\_raspisanie.Items[r]);

MessageBox.Show("Строка номер " + b + " удалена! ", "Авиакасса");

saveToolStripMenu();

tbNumStr.Text = "";

}

После нажатия кнопки управление передается обработчику события btnClearStrNum\_Click\_1. Программа создает переменную строкового типа m и передает в нее значение из свойства Text объекта tbNumStr, далее создается переменная целого типа b, в нее передается методом конвертации значение переменной m. Создается переменная целого типа r, ей присваивается значение b-1. После проверяется есть элементы в lvRaspisanie, если они есть, программа удаляет строку с номером r, в отдельном окне выдается сообщение о том, что строка с номером b удалена. Программа переходит в функцию записи данных в файл.

# **3.1.5.3 Удаление всех строк.**

private void btnClearStrAll\_Click\_1(object sender, EventArgs e)

{

lvRaspisanie.Items.Clear();

MessageBox.Show("Все данные удалены! ", "Авиакасса");

saveToolStripMenu();

}

После нажатия кнопки управление передается обработчику события btnClearStrAll\_Click. Программа с помощью метода Items.Clear() очищает lvRraspisanie, выдает сообщение о том, что все строки удалены и переходит в функцию записи данных в файл.

# **3.1.6. Функция очистки полей информации о пассажире.**

private void btnClear\_Click(object sender, EventArgs e)

{

tbFamily.Text = "Фамилия";

tbName.Text = "Имя";

tbGrazh.Text = "Гражданство";

tbNumDoc.Text = "Серия и № документа";

dtpBithday.Value = System.DateTime.Now;

dtpSrok.Value = System.DateTime.Now;

rtbTiccet.Text = "";

lbCheckReis.Items.Clear();

}

После нажатия кнопки управление передается обработчику события btnClear\_Click. Программа придает соответствующим элементам на форме, значения по tумолчанию.

# **3.2. Схема работы программы.**

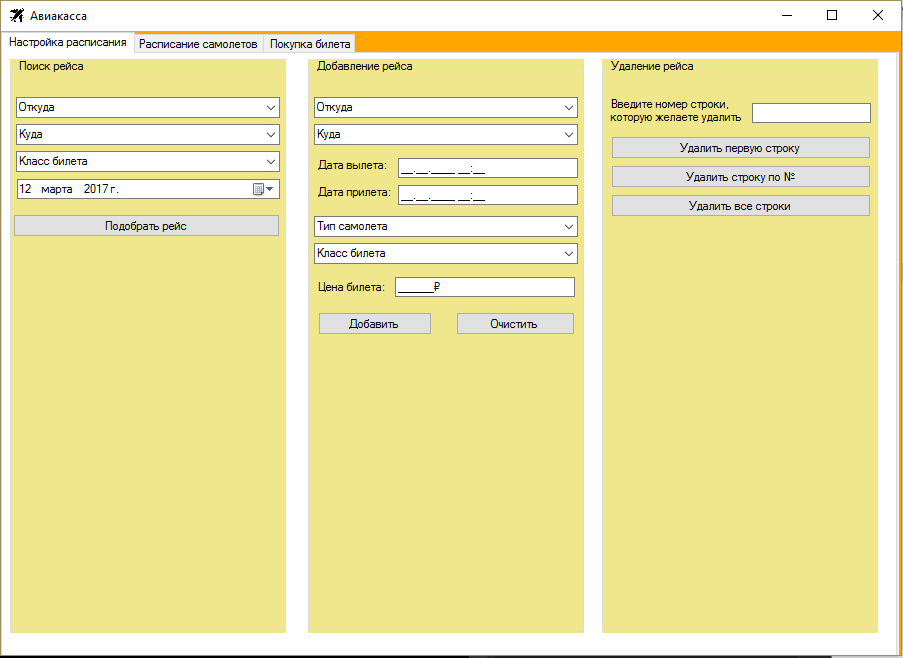
Кассир

Билет

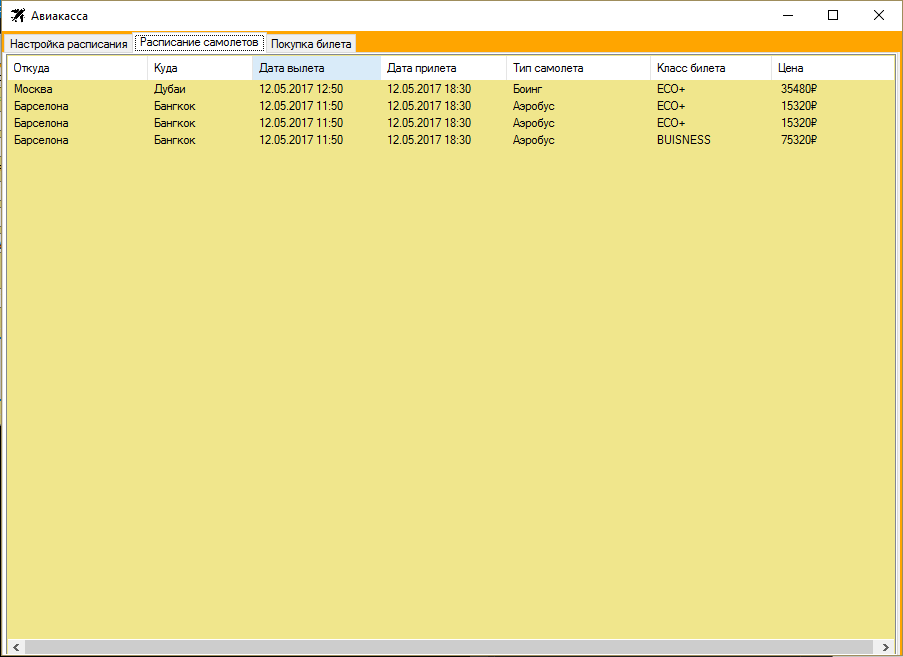
Покупатель

# **3.3. Проектирование интерфейса.**

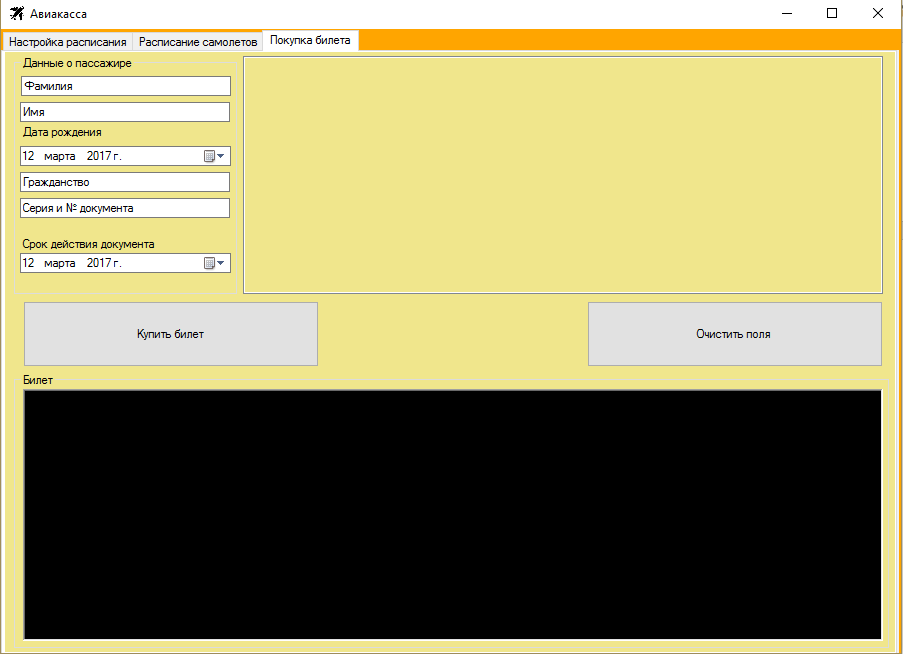
Проектирование интерфейса – не самая простая задач. Интерфейс должен быть интуитивен и понятен не только программисту, создавшему его, но и конечному пользователю, для которого собственно проект и разрабатывается. В ходе разработки интерфейс несколько раз менялся, пока не принял, на мой взгляд, достаточно приятный и понятный для стороннего пользователя вид:



*Рис. 1. Интерфейс программы. Первая вкладка.*



*Рис. 2. Интерфейс программы. Вторая вкладка.*



*Рис. 3. Интерфейс программы. Третья вкладка.*

Первая вкладка – Настройка расписания. Вторая вкладка - Расписание самолетов. Третья вкладка – Покупка билета.

# **Глава 4. Инструкция для пользователя по работе с приложением**

После запуска программы, кассир создает, либо редактирует список рейсов, согласно актуальности, для этого вам надо перейти на вкладку настройка расписания заполнить поля для добавления, либо удаления рейса и нажать соответствующую клавишу в окне приложения. При работе с клиентом, вы узнаете у клиента интересующие его критерии рейса, вносите их в вкладке настройка расписания, окно для поиска рейса. Согласно этим критериям программа подбирает наиболее подходящий рейсы, выделяет их красным цветом и переносит на вкладку покупка билета, где кассир вводит необходимы для покупки данные пользователя и оформляет покупку билета. После покупки файл билета появляется в директории программы и его можно распечатать.

# **Заключение.**

В ходе выполнения этой работы был разработан программный модуль, предоставляющий полный функционал авиакассы. Также были закреплены мои навыки работы с файлами, строками и формами в С#.

# **Список использованных информационных источников:**

1. CLR via C#. Программирование на платформе Microsoft. NET Framework 4.5 на языке C#. 4-е изд. Автор: Дж. Рихтер
2. Язык программирования C#. Классика Computer Science. 4-е изд.

Авторы: А. Хейлсберг, М.Торгерсен, С.Вилтамут, П. Голд

1. Изучаем C#. 3-е изд.

Авторы: Э.Стиллмен, Дж.Грин

1. [www.habrahabr.ru/hub/csharp/](http://www.habrahabr.ru/hub/csharp/) - Хабрахабр, информация по языку C#
2. [https://msdn.microsoft.com/](https://msdn.microsoft.com/ru-RU/library/mt472912(v=vs.110).aspx) - Microsoft Developer Network.
3. Airline Tickets – приложение для покупки авиабилетов на iOS.

# **Приложение 1**

Код программы:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Diagnostics;

using System.IO;

using System.Xml.Linq;

using System.Globalization;

namespace AVIAKASSA1

{

public partial class Aviakassa : Form

{

string UsersDirectory = Environment.CurrentDirectory;

public Aviakassa()

{

InitializeComponent();

parseItems();

}

private void закрытьToolStripMenuItem\_Click(object sender, EventArgs e)

{

Close();

}

bool isWrite = false;

private void saveToolStripMenu()//записывает данные в файл.

{

isWrite = true;

StringBuilder listViewContent = new StringBuilder();

//StringBuilder - Предоставляет изменяемую строку символов.

// ListView - Класс.

//Предоставляет элемент управления списка Windows, отображающий коллекцию элементов, которые могут быть выведены на экран при использовании одного из четырех различных представлений.

//ListViewItem - Инициализирует новый экземпляр класса ListViewItem значениями по умолчанию.

//ListViewItem.ListViewSubItem - Инициализирует новый экземпляр класса ListViewItem.ListViewSubItem значениями по умолчанию.

foreach (ListViewItem item in this.lv\_raspisanie.Items)

{

foreach (ListViewItem.ListViewSubItem subItem in item.SubItems)

{

listViewContent.Append(subItem.Text); // метод класса StringBuilder, добавляет строку к текущей строке.

listViewContent.Append('|');

}

listViewContent.Append(Environment.NewLine);

}

//TextWriter - Представляет модуль записи, который может записывать последовательные наборы символов.Это абстрактный класс.

//StreamWriter - Реализует TextWriter для записи символов в поток в определенной кодировке.

using (TextWriter tw = new StreamWriter(UsersDirectory + "\\airkassa.txt"))

{

tw.WriteLine(listViewContent.ToString()); // Возвращает текущую строку в виде объекта System.String и записывает в файл.

tw.Close();

}

}

private void parseItems()//выводит данные из файла в таблицу ( listView ).

{

lv\_raspisanie.Items.Clear(); // Удаляет все составляющие элементы и столбцы.

using (StreamReader sReader = new StreamReader("airkassa.txt")) //Реализует объект TextReader, который считывает символы из потока байтов в определенной кодировке.

{

string line;

while ((line = sReader.ReadLine()) != null)

{

string[] values = line.Split('|'); // Возвращает строковый массив, содержащий подстроки данного экземпляра, разделенные элементами заданной строки или массива знаков Юникода.

lv\_raspisanie.Items.Add(new ListViewItem(values));

}

}

}

private void Form1\_FormClosing(object sender, FormClosingEventArgs e)//после закрытия формы перезаписывает файл airkass.txt, хранящий в себе информацию о полетах

{

if (isWrite)

{

StreamReader sReader = new StreamReader("airkassa.txt");

string str = sReader.ReadToEnd();

sReader.Close();

TextWriter tw = new StreamWriter(UsersDirectory + "\\airkassa.txt");

tw.Write(str.Remove(str.Length - 4, 4));

tw.Close();

}

}

private void btnAdd\_Click\_1(object sender, EventArgs e)//добавляет рейс в расписание

{

ListViewItem ListItem = new ListViewItem(cbPlusOut.Text);

ListItem.SubItems.Add(cbPlusIn.Text);

ListItem.SubItems.Add(mtbDateOut.Text);

ListItem.SubItems.Add(mtbDateIn.Text);

ListItem.SubItems.Add(cbTypeAir.Text);

ListItem.SubItems.Add(cbKlass.Text);

ListItem.SubItems.Add(mtbPrice.Text);

lv\_raspisanie.Items.Add(ListItem);

MessageBox.Show("Данные добавлены!", "Авиакасс");

saveToolStripMenu();

}

private void btnClearReis\_Click\_1(object sender, EventArgs e)//очищает поля в окне добавления рейса

{

cbPlusOut.Text = "Откуда";

cbPlusIn.Text = "Куда";

cbTypeAir.Text = "Тип самолета";

cbKlass.Text = "Класс билета";

mtbDateOut.Clear();

mtbDateIn.Clear();

mtbPrice.Clear();

MessageBox.Show("Поля очищены!", "Авиакасса");

}

private void btnSeachReis\_Click\_1(object sender, EventArgs e)//осуществляет поиск подходящего покупателю рейса

{

lbCheckReis.Items.Clear();

if (cbSeachOut.Text == string.Empty)

return;

if (cbSeachIn.Text == string.Empty)

return;

if (cbSeachKlass.Text == string.Empty)

return;

string date = dtbDateOut.Value.Date.ToShortDateString();

foreach (ListViewItem item in this.lv\_raspisanie.Items)

{

if ((item.SubItems[0].Text == cbSeachOut.Text) && (item.SubItems[1].Text == cbSeachIn.Text) && (item.SubItems[5].Text == cbSeachKlass.Text) && (item.SubItems[2].Text.Remove(10) == date))

{

item.ForeColor = Color.Red;

lbCheckReis.Items.Add(item.SubItems[0].Text + " " + item.SubItems[1].Text + " " + item.SubItems[2].Text + " " + item.SubItems[3].Text + " " + item.SubItems[4].Text + " " + item.SubItems[5].Text + " " + item.SubItems[6].Text);

}

else

item.ForeColor = Color.Black;

}

}

private void btn\_clearStr1\_Click\_1(object sender, EventArgs e)//удаляет первую строку из расписания

{

if (lv\_raspisanie.Items.Count > 0)

lv\_raspisanie.Items.Remove(lv\_raspisanie.Items[0]);

MessageBox.Show("Первая строка удалена! ", "Авиакасса");

saveToolStripMenu();

}

private void btn\_clearStrNum\_Click\_1(object sender, EventArgs e)//удаляет строку из расписания по номеру введенному в textBox

{

string m = tbNumStr.Text;

int b = Convert.ToInt32(m);

int r = b - 1;

if (lv\_raspisanie.Items.Count > 0)

lv\_raspisanie.Items.Remove(lv\_raspisanie.Items[r]);

MessageBox.Show("Строка номер " + b + " удалена! ", "Авиакасса");

saveToolStripMenu();

tbNumStr.Text = "";

}

private void btn\_clearStrAll\_Click\_1(object sender, EventArgs e)//удаляет все строки из расписания

{

lv\_raspisanie.Items.Clear();

MessageBox.Show("Все данные удалены! ", "Авиакасса");

saveToolStripMenu();

}

private void btnBuyTic\_Click\_1(object sender, EventArgs e)//покупка билета, при нажатии заполняет richTextBox

{

rtbTiccet.Text = "";

StreamReader ticket = File.OpenText(UsersDirectory + "\\ticket.txt");

while (!ticket.EndOfStream)

rtbTiccet.Text += ticket.ReadLine() + "\n";

ticket.Close();

}

private void btnClear\_Click(object sender, EventArgs e)//очистка информации о пассажире и купленном билете

{

tbFamily.Text = "Фамилия";

tbName.Text = "Имя";

tbGrazh.Text = "Гражданство";

tbNumDoc.Text = "Серия и № документа";

dtpBithday.Value = System.DateTime.Now;

dtpSrok.Value = System.DateTime.Now;

rtbTiccet.Text = "";

lbCheckReis.Items.Clear();

}

private void tb\_Family\_MouseDown\_1(object sender, MouseEventArgs e)//очиска поля при нажатии на него мышкой

{

tbFamily.Text = "";

}

private void tb\_Name\_MouseDown\_1(object sender, MouseEventArgs e)//очиска поля при нажатии на него мышкой

{

tbName.Text = "";

}

private void tb\_Grazh\_MouseDown\_1(object sender, MouseEventArgs e)//очиска поля при нажатии на него мышкой

{

tbGrazh.Text = "";

}

private void tb\_NumDoc\_MouseDown\_1(object sender, MouseEventArgs e) //очиска поля при нажатии на него мышкой

{

tbNumDoc.Text = "";

}

private void lb\_checkReis\_MouseDoubleClick\_1(object sender, MouseEventArgs e)//выбор билета для покупки из listBox

{

StreamWriter ticket = File.CreateText(UsersDirectory + "\\ticket.txt");

ticket.WriteLine(" Билет ");

ticket.WriteLine(" Данные о пассажире ");

ticket.WriteLine("Фамилия - " + tbFamily.Text);

ticket.WriteLine("Имя - " + tbName.Text);

ticket.WriteLine("Дата рождения - " + dtpBithday.Value.Date.ToShortDateString());

ticket.WriteLine("Гражданство - " + tbGrazh.Text);

ticket.WriteLine("Серия и № документа - " + tbNumDoc.Text);

ticket.WriteLine("Срок действия документа - " + dtpSrok.Value.Date.ToShortDateString());

ticket.WriteLine(" Информация о полете");

ticket.WriteLine(lbCheckReis.SelectedItem);

ticket.Flush();

ticket.Close();

}

}

}