**МИНОБРНАУКИ РОССИИ**

Федеральное государственное бюджетное

образовательное учреждение высшего образования

**«Ухтинский государственный технический университет»**

**(УГТУ)**

Кафедра вычислительной техники, информационных систем и технологий

**РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА**

Дисциплина: «Объектно-ориентированное программирование»

Шифр: 191407 Группа ИСТ-19о-Б Курс 3

Морданов Егор Владимирович

Проверил:

доцент кафедры ВТИСиТ Кунцев В.Е.

Ухта

2021

содержание

[1 Приложение «Учет авиарейсов» 3](#_Toc91550383)

[2 Шаблон «одиночка». Singleton 11](#_Toc91550384)

[3 Реализация импорт/экспорт в Excel 12](#_Toc91550385)

[4 Модель реализации классов предметной области 15](#_Toc91550386)

[Список использованной литературы 17](#_Toc91550387)

[ПРИЛОЖЕНИЕ 18](#_Toc91550388)

[ПРИЛОЖЕНИЕ 1. Листинг MAIN.cs 19](#_Toc91550389)

[ПРИЛОЖЕНИЕ 2. Листинг frmFlight.cs 22](#_Toc91550390)

[ПРИЛОЖЕНИЕ 3. Листинг frmTicket.cs 23](#_Toc91550391)

[ПРИЛОЖЕНИЕ 4. Листинг frmRegister.cs 24](#_Toc91550392)

[ПРИЛОЖЕНИЕ 5. Листинг frmDeleteTicket.cs 25](#_Toc91550393)

[ПРИЛОЖЕНИЕ 6. Листинг frmSearch.cs 26](#_Toc91550394)

[ПРИЛОЖЕНИЕ 7. Листинг MainManager.cs 28](#_Toc91550395)

# Приложение «Учет авиарейсов»

В рамках работы было создано приложение «Учет авиарейсов».

Главная форма приложения (Рисунок 1) представляет собой окно, разделенное на две части: рейсы и билеты. Листинг формы представлен в Приложении 1.

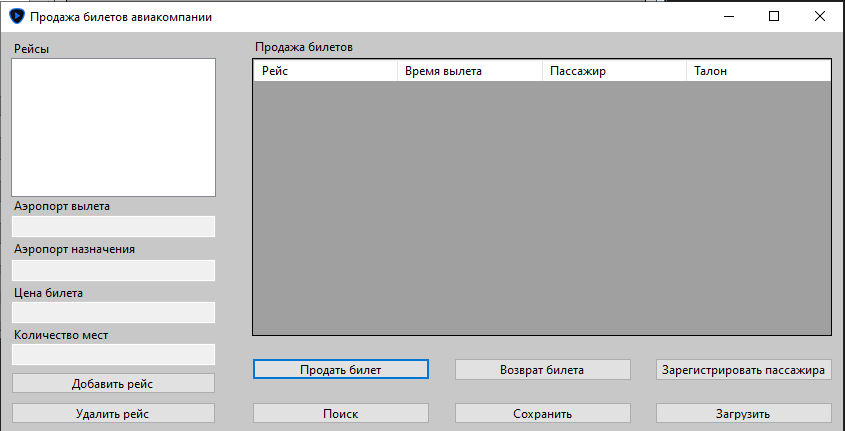


Рисунок - Главная форма

В левой части расположен список всех рейсов, поля, отображающие информацию о выбранном рейсе, кнопки для добавления и удаления рейса.

В правой части расположен список всех проданных билетов, кнопки для продажи, возврата и поиска билетов, регистрации пассажиров, сохранения данных в Excel документ и загрузки из него.

Для создания нового рейса, необходимо нажать на кнопку «Добавить рейс». Откроется форма «Новый рейс» (Рисунок 2). Листинг формы представлен в Приложении 2.

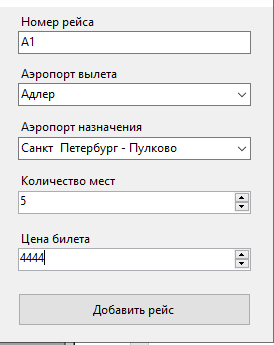


Рисунок - Форма «Новый рейс»

После заполнения информации о рейсе и нажатия кнопки «Добавить рейс», программа вернет пользователя на главную форму и добавит в список созданный рейс (Рисунок 3).

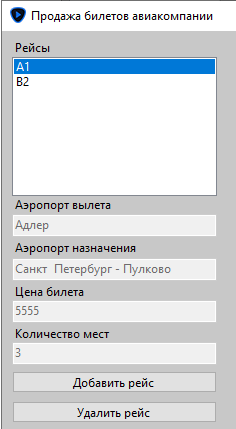


Рисунок - Результат добавления рейса

Для продажи билета, необходимо нажать на кнопку «Продать билет». Откроется форма «Продажа билета» (Рисунок 4). Листинг формы представлен в Приложении 3.

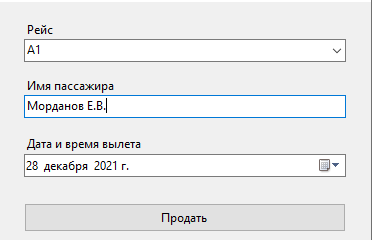


Рисунок – Форма «Продажа билета»

После заполнения информации и нажатия кнопки «Продать», программа вернет пользователя на главную форму и добавит в список проданный билет (Рисунок 5).

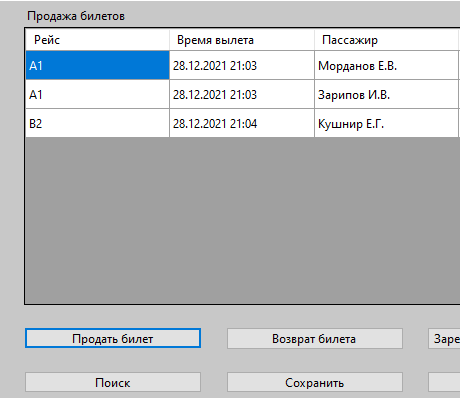


Рисунок - Результат продажи билета

При попытке продать билет на рейс, для которого уже распроданы все билеты программа и не позволит продать билет (Рисунок 6).

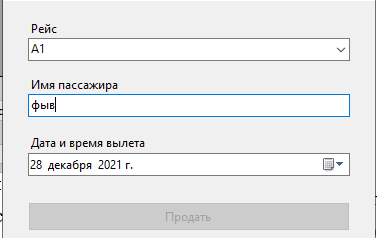


Рисунок - Запрет на продажу билетов

Для регистрации пассажира, необходимо нажать на кнопку «Зарегистрировать пассажира». Откроется форма «Регистрация пассажира» (Рисунок 7). Листинг формы представлен в Приложении 4.

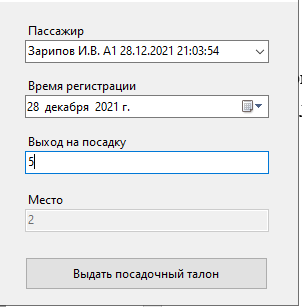


Рисунок – Форма «Регистрация пассажира»

Программа автоматически выбирает доступное место, предоставляемое пассажиру. После заполнения информации и нажатия кнопки «Выдать посадочный талон», программа вернет пользователя на главную форму и добавит в запись о билете информацию о выданном посадочном талоне (Рисунок 8).

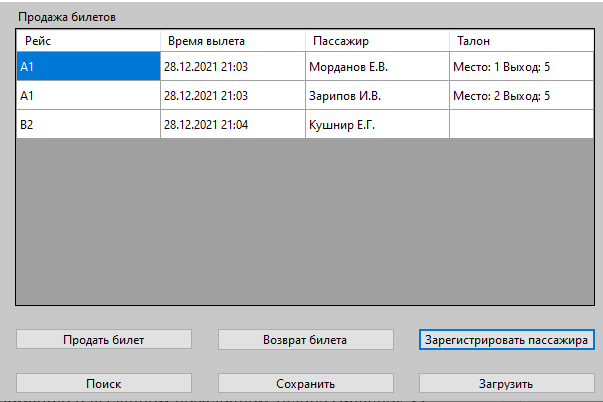


Рисунок - Результат регистрации пассажира

Для удаления билета, необходимо нажать на кнопку «Возврат билета». После этого откроется форма «Возврат билета» (Рисунок 9). Листинг формы представлен в Приложении 5.

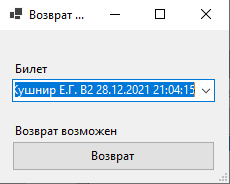


Рисунок 9 – Форма «Возврат билета»

После выбора билета и нажатия кнопки «Возврат» программа вернет пользователя на главную форму и удалит запись о билете из списка (Рисунок 10).

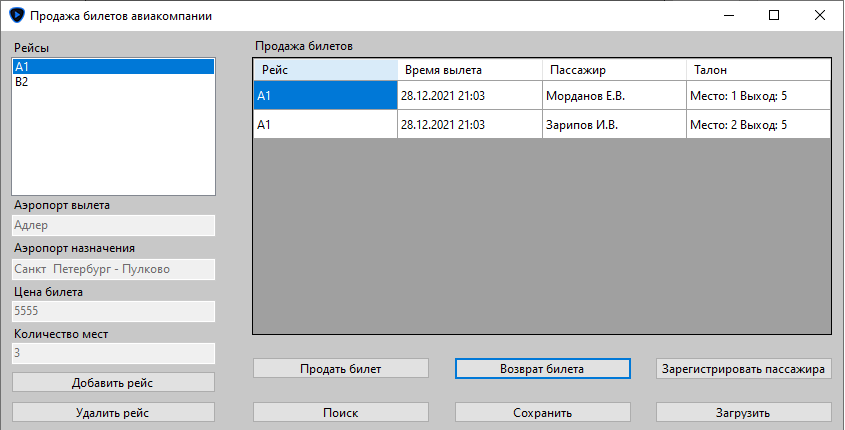


Рисунок 10 - Результат возврата билета

Если пассажир уже зарегистрирован, то возврат будет невозможен (Рисунок 11).

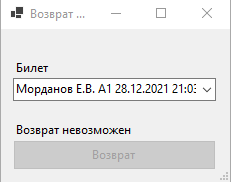


Рисунок 11 - Невозможность возврата билета

Для удаления рейса, необходимо нажать на кнопку «Удалить рейс». После этого будут удалены все билеты на этот рейс из списка билетов и сам рейс (Рисунок 12).

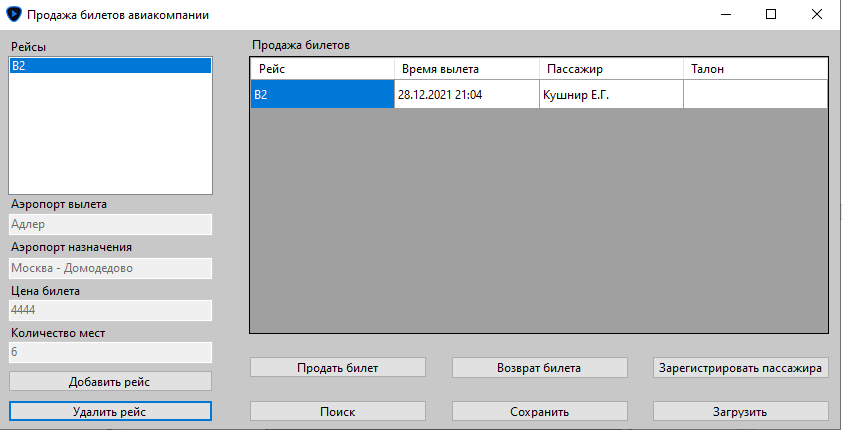


Рисунок 12 - Результат удаления рейса

Для поиска билетов, необходимо нажать на кнопку «Поиск». После этого откроется форма «Поиск проданных билетов» (Рисунок 13). Листинг формы представлен в Приложении 6.

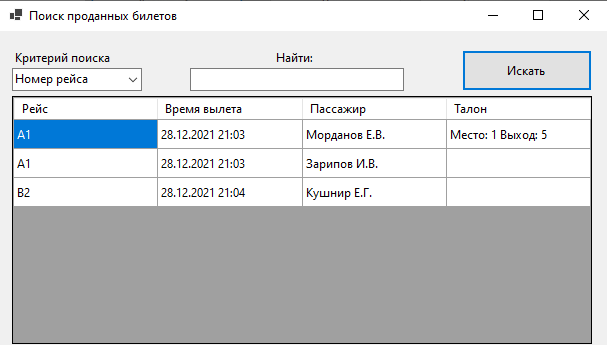


Рисунок 13 - Форма «Поиск проданных билетов»

После выбора критерия поиска, заполнения поля «Найти:» и нажатия кнопки «Поиск» программа выведет на экран список всех проданных билетов, подходящих для выбранного критерия поиска (Рисунок 14).

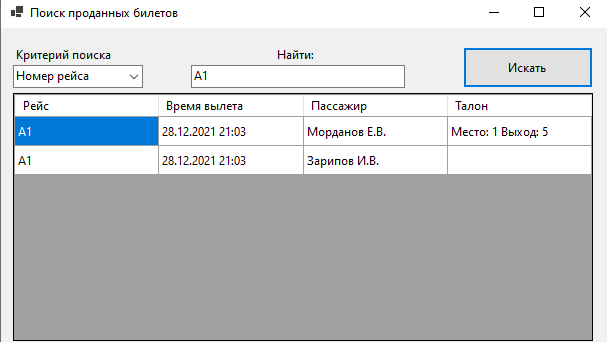


Рисунок 14 - Результат поиска

# Шаблон «одиночка». Singleton

Одиночка ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Singleton) – [порождающий шаблон проектирования](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D0%BE%D0%B6%D0%B4%D0%B0%D1%8E%D1%89%D0%B8%D0%B5_%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D1%8B_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), гарантирующий, что в однопоточном приложении будет единственный экземпляр некоторого класса, и предоставляющий глобальную точку доступа к этому экземпляру.

В данной программе шаблон «одиночка» позволит получать и изменять данные о рейсах и билетах на любой форме без необходимости передачи переменных. Достаточно лишь получить точку доступа к классу «одиночке», в котором находятся все данные.

В классе «одиночке» хранятся списки рейсов и билетов с посадочными талонами, а также процедуры их получения, добавления и удаления, сохранения в файл и загрузки из него.

Скрытый конструктор класса «одиночки» создает пустые списки, готовые к заполнению. Общедоступный конструктор вызывает скрытый, если экземпляр класса еще не был создан, или возвращает ссылку на экземпляр класса, если он уже был создан.

Листинг класса «одиночки» представлен в Приложении 7.

# Реализация импорт/экспорт в Excel

Для реализации импорта/экспорта в Excel требуется библиотека Microsoft.Office.Interop.Excel, перед чем в проекте необходимо добавить ссылку на модель COM Interop.Microsoft.Office.Interop.Excel.

Для экспорта в Excel создана функция SaveToFile(). Данная функция открывает Excel создает новую книгу, после чего на начальном листе в первой строке указывает количество рейсов и билетов, создает «шапку» таблицы. Далее функция переносит в таблицу все записи о рейсах из списка и затем все записи о проданных билетах с их посадочными талонами. В конце функция сохраняет файл с названием «TicketsData.xlsx» в директории программы и закрывает Excel. Работа функции представлена на Рисунке 15.

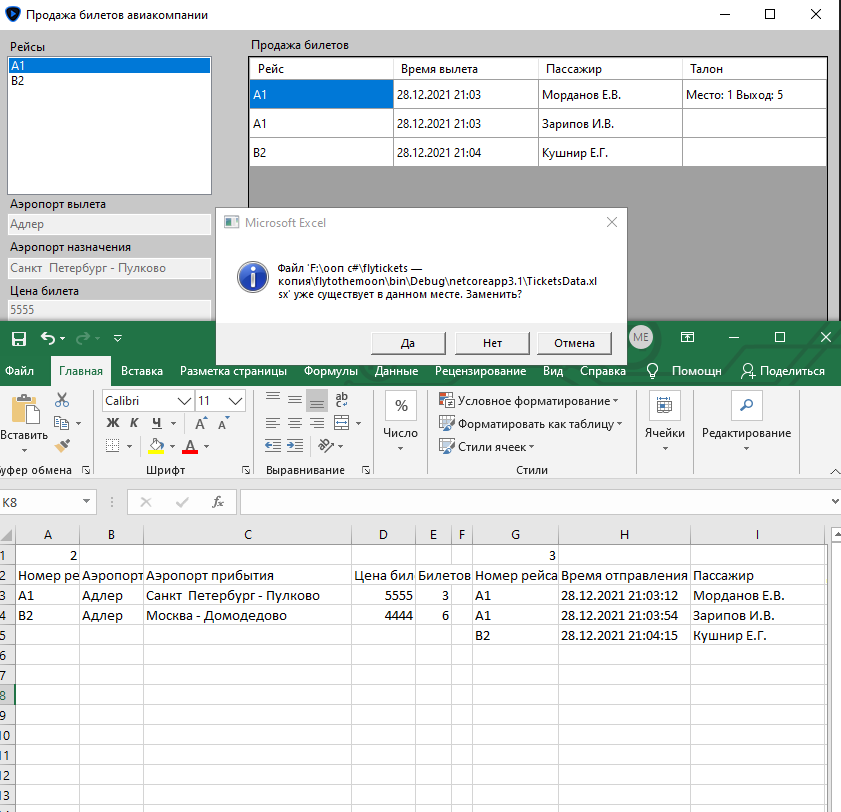
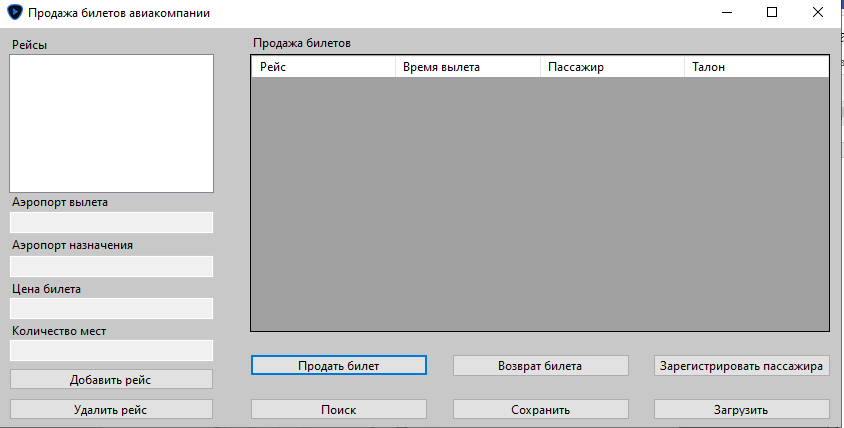


Рисунок 15 - Экспорта в Excel

Для импорта в Excel создана функция LoadFromFile(). Данная функция открывает Excel и в нем файл с названием «TicketsData.xlsx» из директории программы, после чего очищает списки рейсов и билетов в программе. Далее функция переносит из таблицы в программу в соответствующие списки все записи о рейсах и затем все записи о проданных билетах с их посадочными талонами. При этом функция восстанавливает из данных таблицы Excel связи между рейсами, билетами и посадочными талонами. В конце функция закрывает Excel. Работа функции представлена на Рисунке 16.



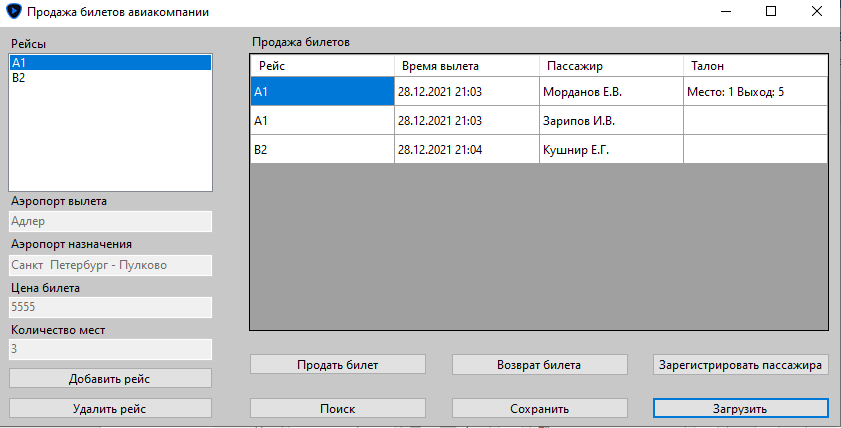


Рисунок 16 - Импорт из Excel

Реализация импорта/экспорта в Excel находится в классе «одиночке», листинг которого представлен в Приложении 7.

# Модель реализации классов предметной области

В предметной области присутствуют такие сущности, как: аэропорт, рейс, билет, посадочный талон. На основе каждой сущности создадим класс.

У класса аэропорт должен быть атрибут «название» и операция преобразования в строку.

У класса рейс должны быть атрибуты: «номер», «пункт отправления», «пункт назначения», «цена билета», «количество билетов», «доступно билетов», «свободные места» (который позволить узнать свободное место и занять его) и операция преобразования в строку. Атрибуты «пункт отправления» и «пункт назначения» имеют тип аэропорт, значит между этими двумя классами есть связь ассоциация. А поскольку разные рейсы могут использовать разные или одни и те же аэропорты и данное использование никак не ограничивается, связь будет многие-ко-многим (аэропорты - рейсы). К каждому рейсу относятся 2 аэропорта, аэропорт может относиться к неограниченному количеству рейсов и не относиться ни к одному.

У класса билет должны быть атрибуты: «рейс», «дата отправления», «пассажир», «посадочный талон» и операция преобразования в строку. Атрибут «рейс» имеет тип рейс, значит между этими двумя классами есть связь ассоциация. А поскольку на один рейс могут быть проданы несколько билетов, связь будет один-ко-многим (рейс - билеты). На один рейс может быть продано любое количество билетов, не превышающее количество мест, или не быть продано ни одного билета. Атрибут «посадочный талон» имеет тип посадочный талон, значит между этими двумя классами есть связь ассоциация. А поскольку на один билет можно получить только один посадочный талон, связь будет один-к-одному (билет – посадочный талон). У одного билета может быть только один посадочный талон или вовсе не быть посадочного талона.

У класса посадочный талон должен быть атрибут «место», «трап», «дата регистрации», «билет» и операция преобразования в строку. Атрибут «билет» имеет тип билет, значит между этими двумя классами есть связь ассоциация. Она уже была обозначена в описании класса билет.

Диаграмма классов предметной области представлена на Рисунке 17.

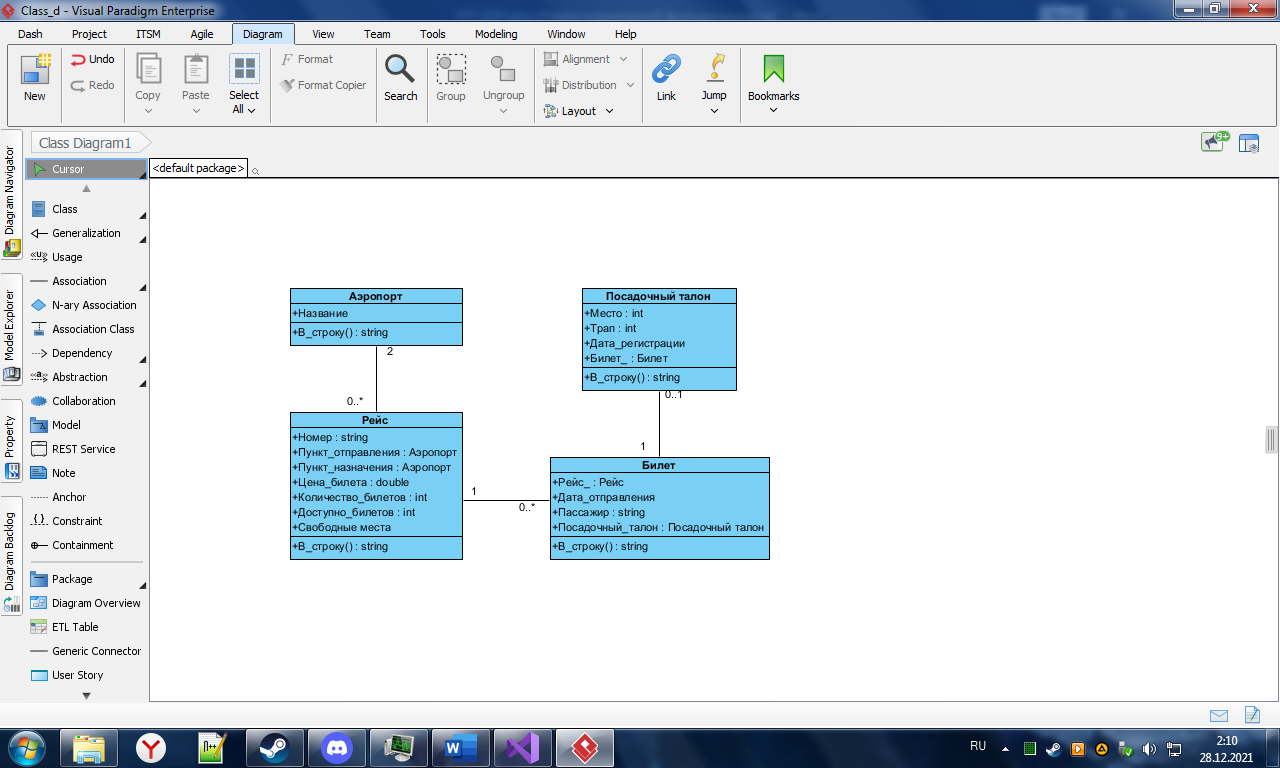


Рисунок 17 - Диаграмма классов предметной области

# Список использованной литературы

1. Куделин, С. Г. Объектные методы. Проектирование и разработка приложений [Текст] : учеб. пособие / С. Г. Куделин, Ф. В. Маракасов, К. В. Рочев. – Ухта: УГТУ, 2018. – 118 с.
2. Кудряшова, О. М. Основы оформления учебных и научных работ с помощью MS Word [Текст] : метод. указания / О. М. Кудряшова. – Ухта : УГТУ, 2016. – 27 с.
3. Ларман, Крэг Применение UML и шаблонов проектирования. 2-е издание. : Пер. с англ. – М. : Издательский дом “Вильямс”, 2004. – 624 с. : ил. – Парал. тит. англ.

# ПРИЛОЖЕНИЕ

# ПРИЛОЖЕНИЕ 1. Листинг MAIN.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  namespace flytothemoon  {  public partial class Main : Form  {  private MainManager mm;  public Main()  {  InitializeComponent();  mm = MainManager.getInstance();  }  private void Main\_Load(object sender, EventArgs e)  {  lbFlights.DisplayMember = "Number";  BindingList<Flight> blF = new BindingList<Flight>(mm.Flights);  lbFlights.DataSource = blF;  dgvTickets.AutoGenerateColumns = false;  dgvTickets.Columns[0].DataPropertyName = "Flight";  dgvTickets.Columns[1].DataPropertyName = "DepartureTime";  dgvTickets.Columns[2].DataPropertyName = "Passenger";  dgvTickets.Columns[3].DataPropertyName = "Pass";  BindingList<Ticket> blT = new BindingList<Ticket>(mm.Tickets);  dgvTickets.DataSource = blT;  ShowFlight();  }  private void ShowFlight()  {  var Buf = lbFlights.SelectedItem as Flight;  if (Buf != null)  {  textBox1.Text = Buf.Departure.ToString();  textBox2.Text = Buf.Arrival.ToString();  textBox3.Text = Buf.Price.ToString();  textBox4.Text = Buf.Tickets.ToString();  }  }  private void lbFlights\_SelectedIndexChanged(object sender, EventArgs e)  {  ShowFlight();  }  private void bAddFlight\_Click(object sender, EventArgs e)  {  frmFlight dlg = new frmFlight();  if (dlg.ShowDialog() == DialogResult.OK) |

Продолжение приложения 1

|  |
| --- |
| MessageBox.Show("Рейс добавлен");  }  BindingList<Flight> blF = new BindingList<Flight>(mm.Flights);  lbFlights.DataSource = blF;  }  private void bSellTicket\_Click(object sender, EventArgs e)  {  frmTicket dlg = new frmTicket();  if (dlg.ShowDialog() == DialogResult.OK)  {  MessageBox.Show("Билет продан");  }  BindingList<Ticket> blT = new BindingList<Ticket>(mm.Tickets);  dgvTickets.DataSource = blT;  }    private void bRegPassenger\_Click(object sender, EventArgs e)  {  frmRegister dlg = new frmRegister();  if (dlg.ShowDialog() == DialogResult.OK)  {  MessageBox.Show("Пассажир зарегистрирован");  }  BindingList<Ticket> blT = new BindingList<Ticket>(mm.Tickets);  dgvTickets.DataSource = blT;  }  private void bDelTicket\_Click(object sender, EventArgs e)  {  frmDeleteTicket dlg = new frmDeleteTicket();  if (dlg.ShowDialog() == DialogResult.OK)  {  MessageBox.Show("Возврат произведен");  }  BindingList<Ticket> blT = new BindingList<Ticket>(mm.Tickets);  dgvTickets.DataSource = blT;  }  private void bDelFlight\_Click(object sender, EventArgs e)  {  Ticket del\_ticket;  do  {  del\_ticket = null;  foreach (Ticket ticket in mm.Tickets)  {  if (ticket.Flight == lbFlights.SelectedItem)  {  del\_ticket = ticket;  }  }  if (del\_ticket != null) mm.Tickets.Remove(del\_ticket);  } while (del\_ticket != null); |

Продолжение приложения 1.

|  |
| --- |
| BindingList<Ticket> blT = new BindingList<Ticket>(mm.Tickets);  dgvTickets.DataSource = blT;  mm.Flights.Remove((Flight)lbFlights.SelectedItem);  BindingList<Flight> blF = new BindingList<Flight>(mm.Flights);  lbFlights.DataSource = blF;  }  private void bSave\_Click(object sender, EventArgs e)  {  try  {  mm.SaveToFile();  MessageBox.Show("Сохранено");  }  catch  {  MessageBox.Show("Не удалось сохранить");  }  }  private void bLoad\_Click(object sender, EventArgs e)  {  try  {  mm.LoadFromFile();  BindingList<Flight> blF = new BindingList<Flight>(mm.Flights);  lbFlights.DataSource = blF;  BindingList<Ticket> blT = new BindingList<Ticket>(mm.Tickets);  dgvTickets.DataSource = blT;  MessageBox.Show("Загружено");  }  catch  {  MessageBox.Show("Не удалось загрузить");  }  }  private void bSearch\_Click(object sender, EventArgs e)  {  frmSearch dlg = new frmSearch();  dlg.Show();  }  }  } |

# ПРИЛОЖЕНИЕ 2. Листинг frmFlight.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Text;  using System.Windows.Forms;  namespace AirFlights  {  public partial class frmFlight : Form  {  private MainManager mm;  public frmFlight()  {  InitializeComponent();  mm = MainManager.getInstance();  cbArrival.DataSource = new List<Airport>(mm.Airports);  cbDeparture.DataSource = new List<Airport>(mm.Airports);  }  private void bOk\_Click(object sender, EventArgs e)  {  Flight newflight = new Flight(tbNum.Text, (Airport)cbDeparture.SelectedItem, (Airport)cbArrival.SelectedItem, (double)nPrice.Value, Convert.ToInt32((double)nSeats.Value));  mm.AddFlight(newflight);  }  }  } |

# ПРИЛОЖЕНИЕ 3. Листинг frmTicket.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Text;  using System.Windows.Forms;  namespace AirFlights  {  public partial class frmTicket : Form  {  private MainManager mm;  public frmTicket()  {  InitializeComponent();  mm = MainManager.getInstance();  cbFlights.DataSource = mm.Flights;  }  private void bOk\_Click(object sender, EventArgs e)  {  Ticket newticket = new Ticket((Flight)cbFlights.SelectedItem, dtpDepartureDate.Value, tbPassenger.Text);  mm.AddTicket(newticket);  }  private void cbFlights\_SelectedIndexChanged(object sender, EventArgs e)  {  Flight Buff = (Flight)cbFlights.SelectedItem;  if (Buff.Tickets\_available <= 0)  {  MessageBox.Show("Все билеты распроданы");  bOk.Enabled = false;  }  else  {  bOk.Enabled = true;  }  }  }  } |

# ПРИЛОЖЕНИЕ 4. Листинг frmRegister.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Text;  using System.Windows.Forms;  namespace AirFlights  {  public partial class frmRegister : Form  {  private MainManager mm;  public frmRegister()  {  InitializeComponent();  mm = MainManager.getInstance();  cbTicket.DataSource = mm.Tickets;  }  private void bOk\_Click(object sender, EventArgs e)  {  new BoardingPass((DateTime)dtpRegtime.Value, Convert.ToInt32((string)tbSeat.Text), Convert.ToInt32((string)tbGate.Text), (Ticket)cbTicket.SelectedItem);  }  private void cbTicket\_SelectedIndexChanged(object sender, EventArgs e)  {  if (((Ticket)cbTicket.SelectedItem).Pass != null)  {  MessageBox.Show("Пассажир уже зарегистрирован");  bOk.Enabled = false;  }  else  {  Ticket buf = mm.Tickets[cbTicket.SelectedIndex];  if(buf!=null) tbSeat.Text = buf.Flight.Seats\_free.ToString();  bOk.Enabled = true;  }  }  }  } |

# ПРИЛОЖЕНИЕ 5. Листинг frmDeleteTicket.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Text;  using System.Windows.Forms;  namespace AirFlights  {  public partial class frmDeleteTicket : Form  {  private MainManager mm;  public frmDeleteTicket()  {  InitializeComponent();  mm = MainManager.getInstance();  cbTickets.DataSource = mm.Tickets;  }  private void cbTickets\_SelectedIndexChanged(object sender, EventArgs e)  {  if (((Ticket)cbTickets.SelectedItem).Pass != null)  {  label2.Text = "Возврат невозможен";  bOk.Enabled = false;  }  else  {  label2.Text = "Возврат возможен";  bOk.Enabled = true;  }  }  private void bOk\_Click(object sender, EventArgs e)  {  mm.Tickets.Remove((Ticket)cbTickets.SelectedItem);  }  }  } |

# ПРИЛОЖЕНИЕ 6. Листинг frmSearch.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Text;  using System.Windows.Forms;  namespace AirFlights  {  public partial class frmSearch : Form  {  private MainManager mm;  private List<string> search\_type = new List<string> {"Номер рейса", "ФИО пассажира", "Дата"};  public frmSearch()  {  InitializeComponent();  mm = MainManager.getInstance();  }  private void frmSearch\_Load(object sender, EventArgs e)  {  cbSearchType.DataSource = search\_type;  dgvTickets.AutoGenerateColumns = false;  dgvTickets.Columns[0].DataPropertyName = "Flight";  dgvTickets.Columns[1].DataPropertyName = "DepartureTime";  dgvTickets.Columns[2].DataPropertyName = "Passenger";  dgvTickets.Columns[3].DataPropertyName = "Pass";  BindingList<Ticket> blT = new BindingList<Ticket>(mm.Tickets);  dgvTickets.DataSource = blT;  }  private void bSearch\_Click(object sender, EventArgs e)  {  try  {  string buf = (string)cbSearchType.SelectedItem;  List<Ticket> result = new List<Ticket>();  foreach (Ticket ticket in mm.Tickets)  {  bool add = false;  if (buf == search\_type[0])  {  if (ticket.Flight.Number == tbSearch.Text) add = true;  }  else if (buf == search\_type[1])  {  if (ticket.Passenger == tbSearch.Text) add = true;  }  else if (buf == search\_type[2])  {  if (ticket.DepartureTime.Date == Convert.ToDateTime(tbSearch.Text).Date) add = true;  }  if (add) result.Add(ticket);  } |

Продолжение приложения 6

|  |
| --- |
| BindingList<Ticket> blT = new BindingList<Ticket>(result);  dgvTickets.DataSource = blT;  }  catch  {  MessageBox.Show("Произошла ошибка. Проверьте введённые данные"  }    }  }  } |

# ПРИЛОЖЕНИЕ 7. Листинг MainManager.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Text;  using ExelLib = Microsoft.Office.Interop.Excel;  using System.Windows.Forms;  namespace AirFlights  {  public class MainManager  {  private static MainManager Instance;  private List<Airport> airports;  private List<Flight> flights;  private List<Ticket> tickets;  private MainManager()  {  airports = new List<Airport>() { new Airport("Адлер"), new Airport("Москва - Домодедово"), new Airport("Санкт\_Петербург - Пулково") };  flights = new List<Flight>();  tickets = new List<Ticket>();  }  public static MainManager getInstance()  {  if (Instance == null)  {  Instance = new MainManager();  }  return Instance;  }  public List<Airport> Airports  {  get { return airports; }  set { airports = (List<Airport>)value; }  }  public bool AddAirport(Airport a)  {  airports.Add(a);  return true;  }  public List<Flight> Flights  {  get { return flights; }  set { flights = (List<Flight>)value; }  }  public bool AddFlight(Flight f)  { |

Продолжение приложения 7

|  |
| --- |
| flights.Add(f);  return true;  }  public bool DelFlight(Flight f)  {  foreach (Ticket ticket in tickets)  {  if (ticket.Flight.Number == f.Number) Tickets.Remove(ticket);  }  flights.Remove(f);  return true;  }  public List<Ticket> Tickets  {  get { return tickets; }  set { tickets = (List<Ticket>)value; }  }  public bool AddTicket(Ticket t)  {  t.Flight.Tickets\_available--;  tickets.Add(t);  return true;  }  public bool DelTicket(Ticket t)  {  t.Flight.Tickets\_available++;  tickets.Remove(t);  return true;  }    public void SaveToFile()  {  string path = System.IO.Directory.GetCurrentDirectory() + @"\TicketsData.xlsx";  ExelLib.Application app = new ExelLib.Application();  ExelLib.Workbook wb = app.Workbooks.Add();  ExelLib.Worksheet ws = wb.ActiveSheet;  //тех. инфо  ws.Cells[1, 1] = Flights.Count;  ws.Cells[1, 7] = Tickets.Count;  //шапка  ws.Cells[2, 1] = "Номер рейса";  ws.Cells[2, 2] = "Аэропорт отправления";  ws.Cells[2, 3] = "Аэропорт прибытия";  ws.Cells[2, 4] = "Цена билета";  ws.Cells[2, 5] = "Билетов";  ws.Cells[2, 7] = "Номер рейса";  ws.Cells[2, 8] = "Время отправления";  ws.Cells[2, 9] = "Пассажир";  ws.Cells[2, 10] = "Дата регистрации";  ws.Cells[2, 11] = "Место";  ws.Cells[2, 12] = "Выход"; |

Продолжение приложения 7

|  |
| --- |
| int row = 3;  foreach (Flight flight in Flights)  {  ws.Cells[row, 1] = flight.Number;  ws.Cells[row, 2] = flight.Departure;  ws.Cells[row, 3] = flight.Arrival;  ws.Cells[row, 4] = flight.Price;  ws.Cells[row, 5] = flight.Tickets;  row++;  }  row = 3;  foreach (Ticket ticket in Tickets)  {  ws.Cells[row, 7] = ticket.Flight.Number;  ws.Cells[row, 8] = ticket.DepartureTime.ToString();  ws.Cells[row, 9] = ticket.Passenger;  if (ticket.Pass != null)  {  ws.Cells[row, 10] = ticket.Pass.RegisterTime.ToString();  ws.Cells[row, 11] = ticket.Pass.Seat;  ws.Cells[row, 12] = ticket.Pass.Gate;  }    row++;  }  app.AlertBeforeOverwriting = false;  wb.SaveAs2(path);  app.Quit();  }  public void LoadFromFile()  {  ExelLib.Application app = new ExelLib.Application();  ExelLib.Workbook wb = app.Workbooks.Open(System.IO.Directory.GetCurrentDirectory() + @"\TicketsData.xlsx");  ExelLib.Worksheet ws = wb.ActiveSheet;  Flights.Clear();  Tickets.Clear();  //создаем список рейсов  int flight\_count = (int)(((ExelLib.Range)ws.Cells[1, 1]).Value2);  for (int row = 3; row <= flight\_count + 2; row++)  {  string number = Convert.ToString( (((ExelLib.Range)ws.Cells[row, 1]).Value2) );  Airport departure = null;  foreach (Airport buf in Airports)  {  if (buf.Name == (string)(((ExelLib.Range)ws.Cells[row, 2]).Value2))  departure = buf;  }  Airport arrival = null;  foreach (Airport buf in Airports)  {  if (buf.Name == (string)(((ExelLib.Range)ws.Cells[row, 3]).Value2))  arrival = buf; |

Продолжение приложения 7

|  |
| --- |
| }  double price = (double)(((ExelLib.Range)ws.Cells[row, 4]).Value2);  int tickets = (int)(((ExelLib.Range)ws.Cells[row, 5]).Value2);  Flight flight = new Flight(number, departure, arrival, price, tickets);  AddFlight(flight);  }  //создаем список билетов  int tickets\_count = (int)(((ExelLib.Range)ws.Cells[1, 7]).Value2);  for (int row = 3; row <= tickets\_count + 2; row++)  {  Flight flight = null;  foreach (Flight buf in Flights)  {  if (buf.Number == Convert.ToString( (((ExelLib.Range)ws.Cells[row, 7]).Value2)) )  flight = buf;  }  DateTime departureTime = Convert.ToDateTime( (((ExelLib.Range)ws.Cells[row, 8]).Value2) );  string passenger = (string)(((ExelLib.Range)ws.Cells[row, 9]).Value2);  Ticket ticket = new Ticket(flight, departureTime, passenger);    if ( (((ExelLib.Range)ws.Cells[row, 10]).Value2!=null) && (((ExelLib.Range)ws.Cells[row, 10]).Value2 !="") )  {  DateTime regtime = Convert.ToDateTime((((ExelLib.Range)ws.Cells[row, 10]).Value2));  int seat = (int)(((ExelLib.Range)ws.Cells[row, 11]).Value2);  int gate = (int)(((ExelLib.Range)ws.Cells[row, 12]).Value2);  new BoardingPass(regtime, seat, gate, ticket);  }    AddTicket(ticket);  }  app.Quit();  }  }  } |