МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образование

учреждение высшего образования

«Ижевский государственный технический университет имени М. Т. Калашникова»

(ФГБОУ ВО «ИжГТУ имени М. Т. Калашникова»)

Институт непрерывного профессионального образования

Лабораторная работа №3

по дисциплине "Прикладное программирование"

(Вариант 2)

|  |  |
| --- | --- |
|  | Выполнил студент группы: 5-19-1 од  Тепляков Георгий Константинович  Принял: преподаватель  Еланцев Михаил Олегович |

Ижевск

2018

**Содержание**

|  |  |
| --- | --- |
| 1 Листинг программы  **1.1 Сheck\_in.cs**  **1.2 DataSource.cs**  **1.3 MemoryDataSource.cs**  **1.4 Name\_client.cs**  **1.5 Nomer.cs**  **1.6 OnlineBooking.cs**  **1.7 TelephoneBooking.cs**  **1.8 ReportItem.cs**  **1.9 Studio.cs**  **1.10 Add.cs**  **1.11 Form1.cs**  **1.12 Program.cs**  **1.13 ReportForm.cs:**  **1.14 UnitTest1.cs:** | **3**  **3**  **3**  **3**  **5**  **5**  **5**  **6**  **6**  **7**  **9**  **12**  **14**  **15**  **16** |
| 2 Контрольный пример | 22 |

1 Листинг программы

**1.1 Сheck\_in.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Accounting\_hotel\_rooms

{

public abstract class Check\_in

{

public int? id;

//public Name\_client ime;

//public Nomer nome;

public string phone;

public string email;

public string name;

public string famil;

public DateTime date;

public int price\_Number;

public int room\_Number;

abstract public Check\_in Copy();

}

}

**1.2 DataSource.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Accounting\_hotel\_rooms

{

public abstract class DataSource

{

abstract public Check\_in Save(Check\_in record);

abstract public bool Delete(int id);

abstract public Check\_in Get(int id, Check\_in record);

abstract public List<Check\_in> GetAll();

}

}

**1.3 MemoryDataSource.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Accounting\_hotel\_rooms

{

public class MemoryDataSource : DataSource

{

private List<Check\_in> check\_ins = new List<Check\_in>();

private int idCounter = 0;

private int FindIndex(int id)

{

for (var i = 0; i < check\_ins.Count; i++)

{

if (check\_ins[i].id == id) return i;

}

return -1;

}

public override Check\_in Get(int id, Check\_in record)

{

var nw = record;

int ind = FindIndex(id);

check\_ins[ind] = nw;

return nw;

}

public override Check\_in Save(Check\_in record)

{

var copy = record.Copy();

if (record.id != null)

{

int ind = FindIndex(record.id.Value);

if (ind != -1)

{

check\_ins[ind] = copy;

return copy;

}

}

copy.id = ++idCounter;

check\_ins.Add(copy);

return copy;

}

public override bool Delete(int id)

{

int ind = FindIndex(id);

if (ind != -1)

{

check\_ins.RemoveAt(ind);

return true;

}

return false;

}

public override List<Check\_in> GetAll()

{

var new\_list = new List<Check\_in>();

foreach (var check in check\_ins)

{

new\_list.Add(check.Copy());

}

return new\_list;

}

}

**1.4 Name\_client.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Accounting\_hotel\_rooms

{

public class Name\_client

{

public string name;

public string famil;

}

}

**1.5 Nomer.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Accounting\_hotel\_rooms

{

public class Nomer

{

public int room\_Number;

public DateTime date;

public int price\_Number;

}

}

**1.6 OnlineBooking.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Accounting\_hotel\_rooms

{

public class OnlineBooking : Check\_in

{

// public string email;

public override Check\_in Copy()

{

return new OnlineBooking

{

id = id,

//ime = ime,

//nome = nome,

email = email,

name = name,

famil = famil,

date = date,

price\_Number = price\_Number,

room\_Number = room\_Number, }; } } }

**1.7 TelephoneBooking.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Accounting\_hotel\_rooms

{

public class TelephoneBooking : Check\_in

{

//public string phone;

public override Check\_in Copy()

{

return new TelephoneBooking

{

id = id,

//ime = ime,

//nome = nome,

phone = phone,

name = name,

famil = famil,

date = date,

price\_Number = price\_Number,

room\_Number = room\_Number,

};

}

}

}

**1.8 ReportItem.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Accounting\_hotel\_rooms

{

public class ReportItem

{

public string strs;

public int type;

public string DisplayingValue

{

get

{

return strs;

}

}

}

}

**1.9 Studio.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Accounting\_hotel\_rooms

{

public class Studio

{

private DataSource dataSource;

public Studio(DataSource data\_source)

{

dataSource = data\_source;

}

/// ----------------------------------------

public Check\_in Save(Check\_in check)

{

if (check is TelephoneBooking)

{

if (check.phone.Length < 15 || check.phone.Length > 15)

{

throw new Exception("Номер веден не верно!");

}

}

else if (check is OnlineBooking)

{

if (check.email.Length < 5 || check.email.Length > 20)

{

throw new Exception("Email веден не верно!");

}

}

if (check.name == null || check.name.Length < 1)

{

throw new Exception("Имя отсутствует!");

}

if (check.famil == null || check.famil.Length < 1)

{

throw new Exception("Фамилия отсутствует!");

}

if (check.room\_Number < 1 || check.room\_Number > 4)

{

throw new Exception("Выберите тип номера");

}

return dataSource.Save(check);

}

/// ----------------------------------------

public bool Delete(int id)

{

if (id < 1)

{

throw new Exception(" Отсутствует id!");

}

return dataSource.Delete(id);

}

/// ----------------------------------------

public List<Check\_in> GetList()

{

var check = dataSource.GetAll();

check.Sort((a, b) =>

{

if (a.date < b.date) return -1;

else if (a.date > b.date) return 1;

else

{

if (a.room\_Number < b.room\_Number) return -1;

else if (a.room\_Number > b.room\_Number) return 1;

else return 0;

}

});

return check;

}

public List<ReportItem> Report(DateTime? dataRep = null)

{

var re = dataSource.GetAll();

int rep = 0;

string rea = " ";

string str = "";

var s = new ReportItem();

s.type = 1;

foreach (Check\_in r in re)

{

if (dataRep == null || r.date == dataRep.Value)

{

if (r is TelephoneBooking) rea = "номер\_телефона:" + ((TelephoneBooking)r).phone;

else if (r is OnlineBooking) rea = "email:" + ((OnlineBooking)r).email;

str = str + "+" + r.price\_Number + " "

+ "Дата заезда:" + r.date.ToString("d MMMM yyyy") + " "

+ "№\_номера:" + r.room\_Number + " "

+ "Имя:" + r.name + " "

+ "Фамилия:" + r.famil + " "

+ rea + " "

+ "id:" + r.id + ' '

+ " " + Environment.NewLine;

rep += r.price\_Number;

}

}

str += Environment.NewLine + Environment.NewLine + "Итог за:" + " " + rep + "р" ;

s.strs += str;

return new List<ReportItem>

{

s,

};

}

}

}

**1.10 Add.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;

using Accounting\_hotel\_rooms;

namespace Accounting\_hotel\_rooms\_Windows

{

public partial class Add : Form

{

public Check\_in Editing { get; set; }

public Add()

{

InitializeComponent();

Box\_phone.Enabled = false;

Box\_email.Enabled = false;

}

private void Type\_SelectedIndexChanged(object sender, EventArgs e)

{

if (Type.SelectedIndex == 0)

{

Box\_phone.Enabled = false;

Box\_email.Enabled = true;

}

else

{

Box\_phone.Enabled = true;

Box\_email.Enabled = false;

}

}

private void Button1\_Click(object sender, EventArgs e)

{

Close();

}

private void Box\_name\_MaskInputRejected(object sender, MaskInputRejectedEventArgs e)

{

Box\_name.SelectionStart = Box\_name.MaskedTextProvider.LastAssignedPosition + 1;

Box\_name.SelectionLength = 0;

}

private void Box\_familya\_MaskInputRejected(object sender, MaskInputRejectedEventArgs e)

{

Box\_familya.SelectionStart = Box\_familya.MaskedTextProvider.LastAssignedPosition + 1;

Box\_familya.SelectionLength = 0;

}

private void maskedphone\_MaskInputRejected\_1(object sender, MaskInputRejectedEventArgs e)

{

maskedphone.SelectionStart = maskedphone.MaskedTextProvider.LastAssignedPosition + 1;

maskedphone.SelectionLength = 0;

}

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

{

if (Type.SelectedIndex == -1)

{

MessageBox.Show("Выберите тип регистрации");

return;

}

Check\_in check;

if (Type.SelectedIndex == 0)

{

check = new OnlineBooking();

((OnlineBooking)check).email = maskedemail.Text;

//if (maskedemail.Text.Length < 5 || maskedemail.Text.Length > 20)

//{

// MessageBox.Show("Email веден не верно!");

// return;

//}

}

else

{

check = new TelephoneBooking();

((TelephoneBooking)check).phone = maskedphone.Text;

//if (maskedphone.Text.Length < 15 || maskedphone.Text.Length > 15)

//{

// MessageBox.Show("Номер веден не верно!");

// return;

//}

}

//if (Box\_name.Text.Length < 1)

//{

// MessageBox.Show("Имя отсутствует!");

// return;

//}

//if (Box\_familya.Text.Length < 1)

//{

// MessageBox.Show("Фамилия отсутствует!");

// return;

//}

//if (Type\_nomer.SelectedIndex == -1)

//{

// MessageBox.Show("Выберите тип номера");

// return;

//}

check.id = Editing != null ? Editing.id : null;

check.email = maskedemail.Text;

check.phone = maskedphone.Text;

//check.ime.name = Box\_name.Text;

//check.ime.famil = Box\_familya.Text;

//check.nome.room\_Number = Type\_nomer.SelectedIndex + 1;

//check.nome.date = dateTimePicker1.ToString();

check.room\_Number = Type\_nomer.SelectedIndex + 1;

check.name = Box\_name.Text;

check.famil = Box\_familya.Text;

check.date = this.dateTimePicker1.Value.Date;

if (Type\_nomer.SelectedIndex == +0)

{

check.price\_Number = 1000;

}

else if (Type\_nomer.SelectedIndex == +1)

{

check.price\_Number = 1200;

}

else if (Type\_nomer.SelectedIndex == +2)

{

check.price\_Number = 1600;

}

else if (Type\_nomer.SelectedIndex == +3)

{

check.price\_Number = 2500;

}

try

{

Program.Studio.Save(check);

Close();

}

catch (Exception err)

{

MessageBox.Show(err.Message);

}

}

private void Label3\_Click(object sender, EventArgs e) {/\* выберите тип регистрции \*/}

private void Label1\_Click(object sender, EventArgs e) {/\* введите номер \*/ }

private void Label2\_Click(object sender, EventArgs e) {/\* введите email \*/}

private void Label6\_Click(object sender, EventArgs e) {/\* введите имя \*/}

private void Label7\_Click(object sender, EventArgs e) {/\* введите фамилию\*/}

private void Label4\_Click(object sender, EventArgs e) {/\*выберите номер\*/ }

private void Label5\_Click(object sender, EventArgs e) {/\*выберите дату заселения\*/}

private void Maskedemail\_MaskInputRejected(object sender, MaskInputRejectedEventArgs e) { }

private void Add\_Load(object sender, EventArgs e) { }

private void Type\_nomer\_SelectedIndexChanged(object sender, EventArgs e) { }

private void dateTimePicker1\_ValueChanged(object sender, EventArgs e) { }

}

}

**1.11 Form1.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;

using Accounting\_hotel\_rooms;

namespace Accounting\_hotel\_rooms\_Windows

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

RefreshTable();

}

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

{

var add = new Add();

add.ShowDialog();

RefreshTable();

}

public void RefreshTable()

{

Tableaddreport.Controls.Clear();

Tableaddreport.RowStyles.Clear();

var checks = Program.Studio.GetList();

Tableaddreport.RowCount = checks.Count;

for (var i = 0; i < checks.Count; i++)

{

var check = checks[i];

var idLabel = new Label();

idLabel.Text = "id:" + " " + check.id.ToString();

idLabel.Width = 45;

Tableaddreport.Controls.Add(idLabel, 0, i);

var data = new Label();

//data = checks.nome.date;

data.Text = "Дата заезда:" + " " + check.date.ToString("d MMMM yyyy");

data.Width = 235;

Tableaddreport.Controls.Add(data, 1, i);

var nomeLabel = new Label();

//nomeLabel.Text = checks.nome.room\_Number.ToString();

nomeLabel.Text = "№\_номера:" + " " + check.room\_Number.ToString();

nomeLabel.Width = 100;

Tableaddreport.Controls.Add(nomeLabel, 2, i);

var nomes = new Label();

//nomes.Text = " " + checks.nome.price\_Number + "р".ToString();

nomes.Text = "+" + check.price\_Number + "р".ToString();

nomes.Width = 70;

Tableaddreport.Controls.Add(nomes, 3, i);

var e\_m = new Label();

e\_m.Text = check is OnlineBooking? "Email:" + " " + check.email : "Телефон:" + " " + check.phone;

e\_m.Width = 220;

Tableaddreport.Controls.Add(e\_m, 4, i);

var nameLabel = new Label();

// nameLabel.Text = checks.ime.name.ToString();

nameLabel.Text = "Имя:" + " " + check.name.ToString();

nameLabel.Width = 150;

Tableaddreport.Controls.Add(nameLabel, 5, i);

var famLabel = new Label();

// nameLabel.Text = checks.ime.famil.ToString();

famLabel.Text = "Фамилия:" + " " + check.famil.ToString();

famLabel.Width = 240;

Tableaddreport.Controls.Add(famLabel, 6, i);

///----------------------------------------

var izmButton = new Button();

izmButton.Text = "Изменить";

izmButton.Click += (object sender, EventArgs e) =>

{

var add = new Add();

add.Editing = check;

add.ShowDialog(); //показать

RefreshTable(); // обновить таблицу

};

Tableaddreport.Controls.Add(izmButton, 7, i);

izmButton.Width = 90;

izmButton.Height = 27;

///----------------------------------------

var deleteButton = new Button();

deleteButton.Text = "Удалить";

deleteButton.Click += (object sender, EventArgs e) =>

{

var button = MessageBox.Show("Вы уверены что хотите удалить?", "Подтверждение", MessageBoxButtons.YesNo);

if (button == DialogResult.Yes)

{

Program.Studio.Delete(check.id.Value);

RefreshTable();

}

};

deleteButton.Width = 90;

deleteButton.Height = 27;

Tableaddreport.Controls.Add(deleteButton, 8, i);

Tableaddreport.RowStyles.Add(new RowStyle()

{

SizeType = SizeType.AutoSize

}

);

}

}

///отчет

private void To\_form\_ToolStripMenuItem\_Click(object sender, EventArgs e)

{

var reportForm = new ReportForm();

reportForm.ShowDialog();

}

///кнопка выход

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

{

Application.Exit();

}

private void MenuToolStripMenuItem\_Click(object sender, EventArgs e) { }

private void Form1\_Load(object sender, EventArgs e) { }

private void Tableaddreport\_Paint(object sender, PaintEventArgs e) { }

}

}

**1.12 Program.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

using Accounting\_hotel\_rooms;

namespace Accounting\_hotel\_rooms\_Windows

{

static class Program

{

public static Studio Studio;

[STAThread]

static void Main()

{

Studio = new Studio(new MemoryDataSource());

/\*Studio.Save(new TelephoneBooking

{

phone = "8(950)836–46–92",

room\_Number = 1 ,

name = "sdbbd",

famil = "asdbd",

price\_Number= 1000,

});

Studio.Save(new OnlineBooking

{

email = "cold@mail.ru",

room\_Number = 1,

name = "sdbbd",

famil = "asdbd",

price\_Number = 1000,

});\*/

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

} }

**1.13 ReportForm.cs:**

using Accounting\_hotel\_rooms;

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.IO;

using System.Windows.Forms;

namespace Accounting\_hotel\_rooms\_Windows

{

public partial class ReportForm : Form

{

public ReportForm()

{

InitializeComponent();

}

private List<ReportItem> Generate()

{

if (Type.SelectedIndex == -1)

{

MessageBox.Show("Выберите тип отчета");

}

if (Type.SelectedIndex == +0)

{

return Program.Studio.Report(null);

}

else if (Type.SelectedIndex == +1)

{

DateTime dataRep = this.dateTimePicker2.Value.Date;

return Program.Studio.Report(dataRep);

}

return null;

}

private void button2\_Click(object sender, EventArgs e)

{

listBox1.DataSource = Generate();

}

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

{

var res = saveFileDialog1.ShowDialog();

if (res == DialogResult.OK)

{

var report = Generate();

using (var stream = new FileStream(saveFileDialog1.FileName, FileMode.CreateNew))

using (var writer = new StreamWriter(stream))

{

foreach (var r in report)

{

writer.WriteLine(r.DisplayingValue);

}

}

}

}

private void button5\_Click(object sender, EventArgs e)

{

Close();

}

private void ReportForm\_Load(object sender, EventArgs e) { }

private void label1\_Click(object sender, EventArgs e) { }

private void groupBox2\_Enter(object sender, EventArgs e) { }

private void listBox1\_SelectedIndexChanged(object sender, EventArgs e) {}

private void saveFileDialog1\_FileOk(object sender, CancelEventArgs e) { }

private void dateTimePicker2\_ValueChanged(object sender, EventArgs e) { }

private void Type\_nomer\_SelectedIndexChanged(object sender, EventArgs e)

{

if (Type.SelectedIndex == +0)

{

groupBox3.Enabled = false;

}

else if (Type.SelectedIndex == +1)

{

groupBox3.Enabled = true;

}

else if (Type.SelectedIndex == -1)

{

groupBox3.Enabled =false;

}

}

}

}

**1.14 UnitTest1.cs:**

using System;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using Accounting\_hotel\_rooms;

namespace UnitTest

{

[TestClass]

public class UnitTest1

{

[TestMethod]

public void TestAdd()

{

var memory = new MemoryDataSource();

var studio = new Studio(memory);

var check = new OnlineBooking

{

email = "cold@mail.ru",

name = "rthrth",

famil = "rthfgnt",

room\_Number = 1,

};

var test = studio.Save(check);

var checks = new TelephoneBooking

{

phone = "8(950)836–46–92",

name = "rthrth",

famil = "rthfgnt",

room\_Number = 1,

};

var tests = studio.Save(checks);

var list = studio.GetList();

Assert.AreEqual(list.Count, 2);

Assert.AreEqual(list[0].phone, check.phone);

Assert.AreEqual(list[0].name, check.name);

Assert.AreEqual(list[0].famil, check.famil);

Assert.AreEqual(list[0].room\_Number, check.room\_Number);

Assert.AreEqual(test.id, 1);

Assert.AreEqual(list[1].email, checks.email);

Assert.AreEqual(list[1].name, checks.name);

Assert.AreEqual(list[1].famil, checks.famil);

Assert.AreEqual(list[1].room\_Number, checks.room\_Number);

Assert.AreEqual(tests.id, 2);

}

[TestMethod]

public void TestError()

{

var memory = new MemoryDataSource();

var studio = new Studio(memory);

var check = new OnlineBooking

{

email = "",

name = "rthrth",

famil = "rthfgnt",

room\_Number = 1,

};

Assert.ThrowsException<Exception>(() =>

{

studio.Save(check);

}, "Email веден не верно!");

var check2 = new OnlineBooking

{

email = "cold@mail.ru",

name = "",

famil = "rthfgnt",

room\_Number = 1,

};

Assert.ThrowsException<Exception>(() =>

{

studio.Save(check2);

}, "Имя отсутствует!");

var check3 = new OnlineBooking

{

email = "cold@mail.ru",

name = "rthrth",

famil = "",

room\_Number = 1,

};

Assert.ThrowsException<Exception>(() =>

{

studio.Save(check3);

}, "Фамилия отсутствует!");

var check4 = new OnlineBooking

{

email = "cold@mail.ru",

name = "rthrth",

famil = "rthfgnt",

room\_Number = 0,

};

Assert.ThrowsException<Exception>(() =>

{

studio.Save(check4);

}, "Выберите тип номера");

var checks = new TelephoneBooking

{

phone = "",

name = "rthrth",

famil = "rthfgnt",

room\_Number = 1,

};

Assert.ThrowsException<Exception>(() =>

{

studio.Save(checks);

}, "Номер веден не верно!");

var check5 = new TelephoneBooking

{

phone = "8(950)836–46–92",

name = "",

famil = "rthfgnt",

room\_Number = 1,

};

Assert.ThrowsException<Exception>(() =>

{

studio.Save(check2);

}, "Имя отсутствует!");

var check6 = new TelephoneBooking

{

phone = "8(950)836–46–92",

name = "rthrth",

famil = "",

room\_Number = 1,

};

Assert.ThrowsException<Exception>(() =>

{

studio.Save(check3);

}, "Фамилия отсутствует!");

var check7 = new TelephoneBooking

{

phone = "8(950)836–46–92",

name = "rthrth",

famil = "rthfgnt",

room\_Number = 0,

};

Assert.ThrowsException<Exception>(() =>

{

studio.Save(check4);

}, "Выберите тип номера");

}

[TestMethod]

public void TestUpdate()

{

var memory = new MemoryDataSource();

var studio = new Studio(memory);

var check = new OnlineBooking

{

email = "cold@mail.ru",

name = "rthrth",

famil = "rthfgnt",

room\_Number = 1,

};

var test = studio.Save(check);

test.email = "colfdd@mail.ru";

test.name = "cold";

test.famil = "colds";

test.room\_Number = 3;

studio.Save(test);

var checks = new TelephoneBooking

{

phone = "8(950)836–46–92",

name = "rthrth",

famil = "rthfgnt",

room\_Number = 1,

};

var tests = studio.Save(checks);

tests.phone = "8(950)836–46–93";

tests.name = "cold";;

tests.famil = "colds";

tests.room\_Number = 3;

studio.Save(tests);

var list = studio.GetList();

Assert.AreEqual(list.Count, 2);

Assert.AreEqual(list[0].email, "colfdd@mail.ru");

Assert.AreEqual(list[0].name, "cold");

Assert.AreEqual(list[0].famil, "colds");

Assert.AreEqual(list[0].room\_Number, 3);

Assert.AreEqual(test.id, 1);

Assert.AreEqual(list[1].phone, "8(950)836–46–93");

Assert.AreEqual(list[1].name, "cold");

Assert.AreEqual(list[1].famil, "colds");

Assert.AreEqual(list[1].room\_Number, 3);

Assert.AreEqual(tests.id, 2);

}

[TestMethod]

public void TestDelete()

{

var memory = new MemoryDataSource();

var studio = new Studio(memory);

var pencil = new TelephoneBooking

{

phone = "8(950)836–46–92",

name = "rthrth",

famil = "rthfgnt",

room\_Number = 1,

};

var new\_pencil = studio.Save(pencil);

int ids = new\_pencil.id;

studio.Delete(ids);

var list = studio.GetList();

Assert.AreEqual(new\_pencil.id, 1);

}

[TestMethod]

public void TestReport()

{

var memory = new MemoryDataSource();

var studio = new Studio(memory);

studio.Save(new TelephoneBooking

{

phone = "8(950)836–46–92",

room\_Number = 1,

name = "sdbbd",

famil = "asdbd",

price\_Number = 1000,

});

studio.Save(new OnlineBooking

{

email = "cold@mail.ru",

room\_Number = 1,

name = "sdbbd",

famil = "asdbd",

price\_Number = 1000,

});

var re = studio.Report();

foreach (var r in re)

{

string str = "";

Assert.AreNotEqual(r.strs, str);

}

}

}

}

2 Контрольный пример

На рисунке 1 показано главное меню.

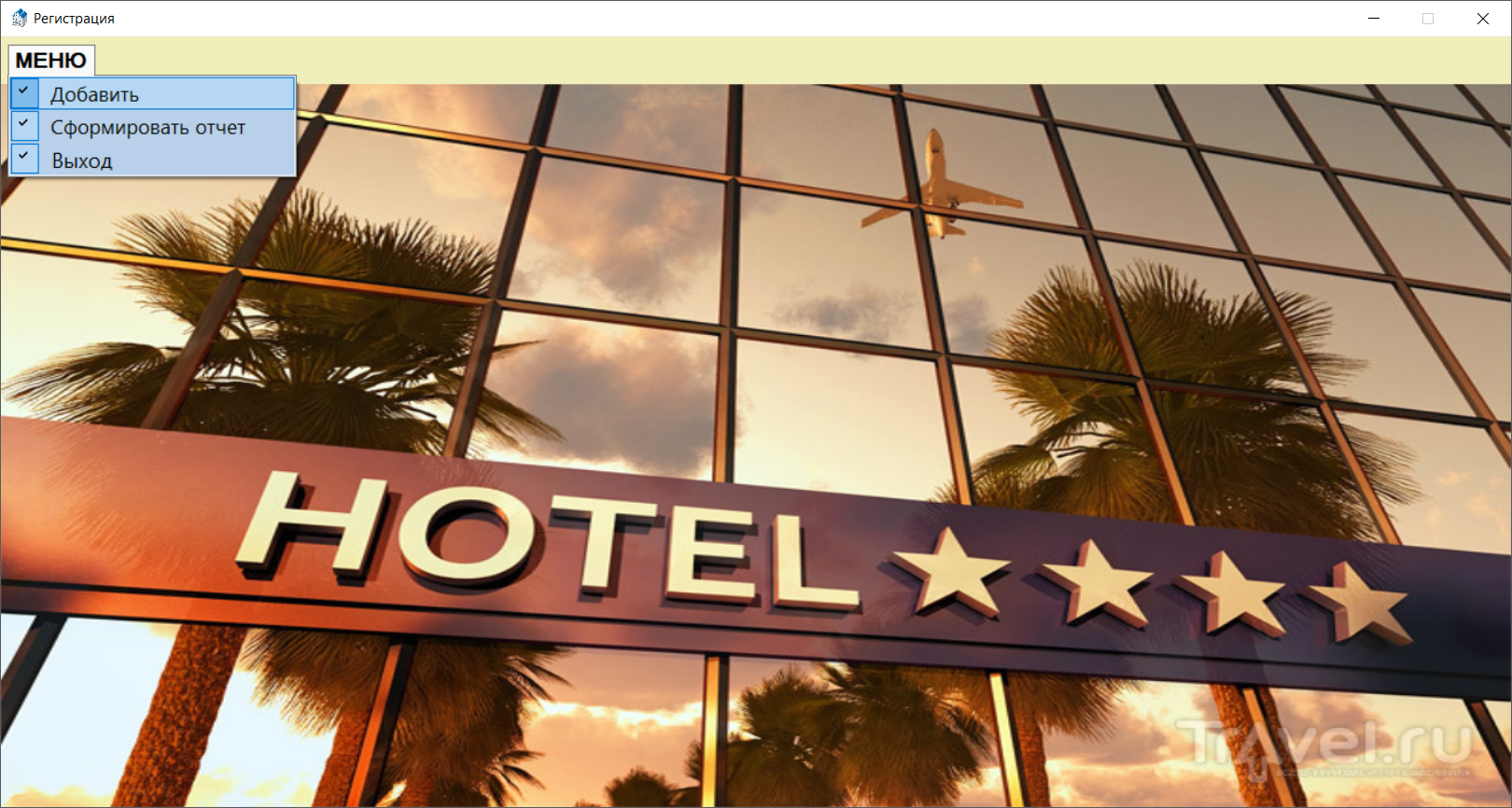


Рис. 1 – главное меню

На рисунке 2 показана форма добавление новой записи.

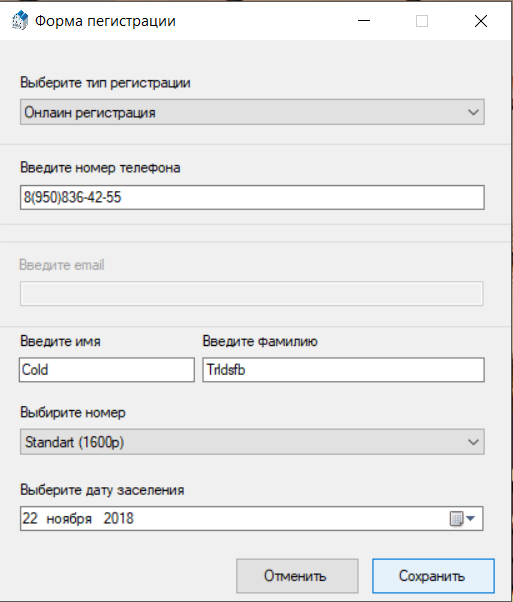


Рис. 2 – форма добавление новой записи

На рисунке 3 представлен вывод всех записей на текущий момент в главном окне программы.

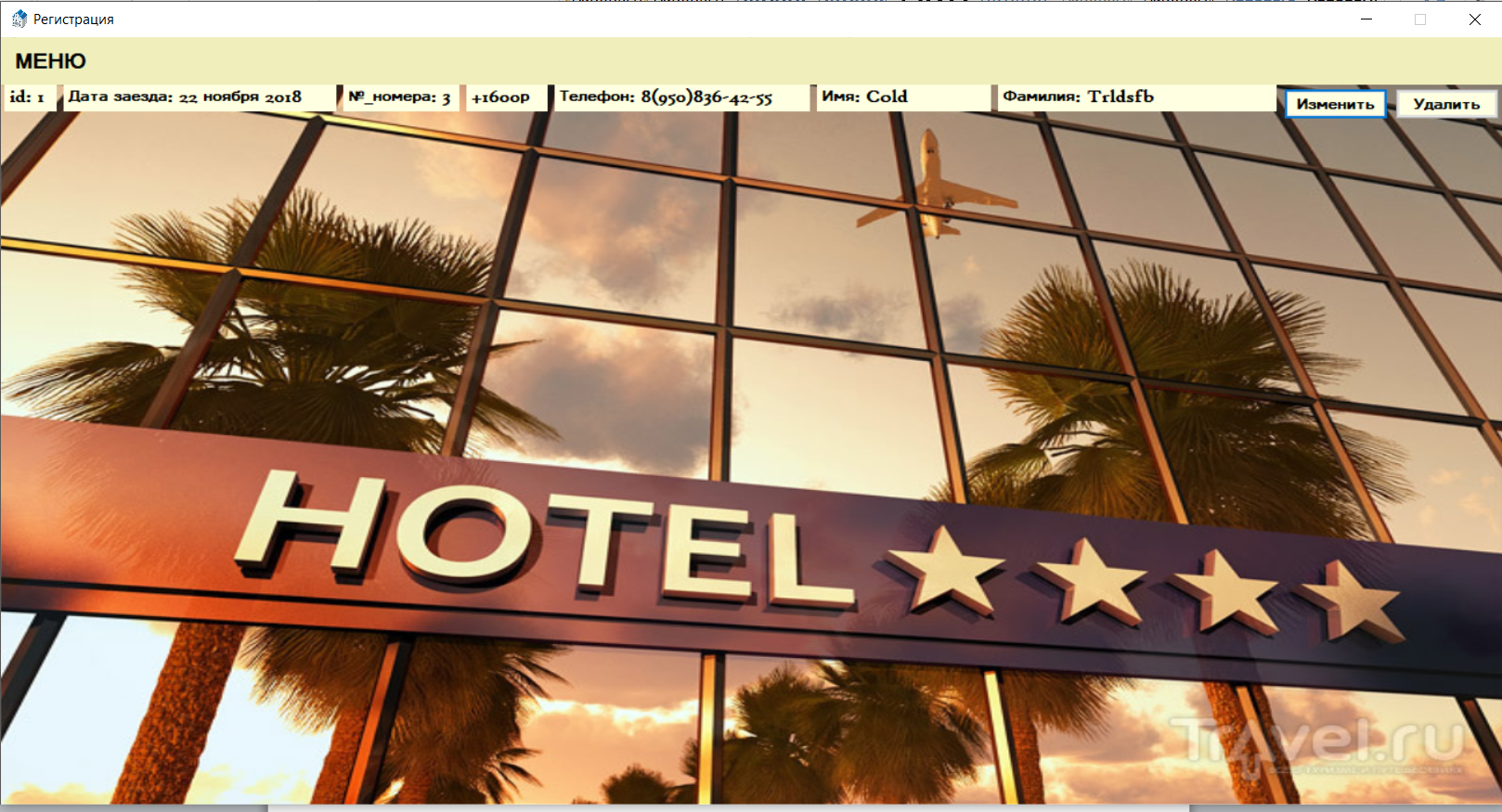


Рис. 3 – вывод всех записей на текущий момент в главном окне программы

Аналогично добавим еще несколько записей. После добавления на рисунке 4 происходит сортировка по полям с датой заезда и №\_номера.



Рис. 4 – сортировка записей на текущий момент

На рисунке 5 показано изменение записи с id = 1.

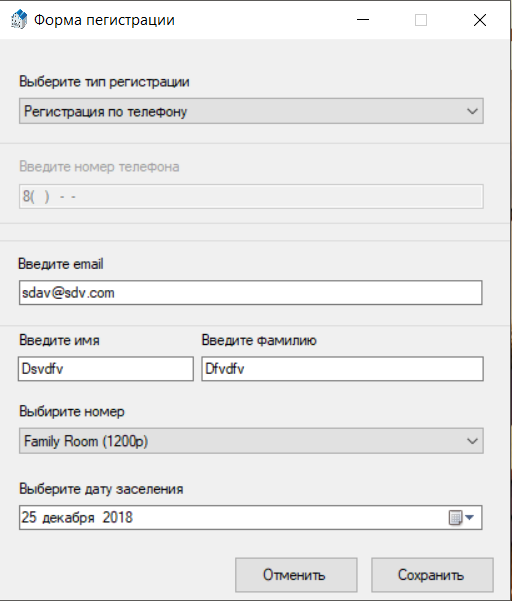


Рис. 5 – изменение записи с id = 1

На рисунке 6 показана запись до изменения и сравнение ее с рисунком 7 после изменение записи с id = 1.

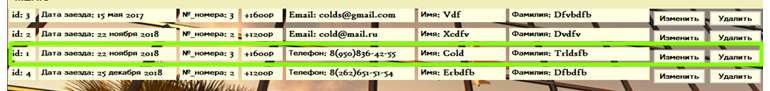


Рис. 6 – запись до изменения с id = 1



Рис. 7– запись после изменение с id = 1

На рисунке 8 показано удаление записи с id = 1.



Рис. 8 – удаление записи с id = 1

На рисунке 6 показана запись до удаления и сравнение ее с рисунком 9 после изменение записи с id = 1.

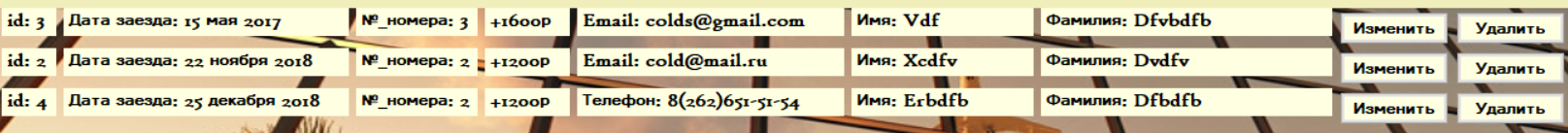


Рис. 9 – вид после удаления записи с id = 1

На рисунке 10 был создан и выведен отчёт за все время. Сам отчёт сохранен в виде .txt файла, содержащий в себе все имеющиеся записи и подсчет их сумму.

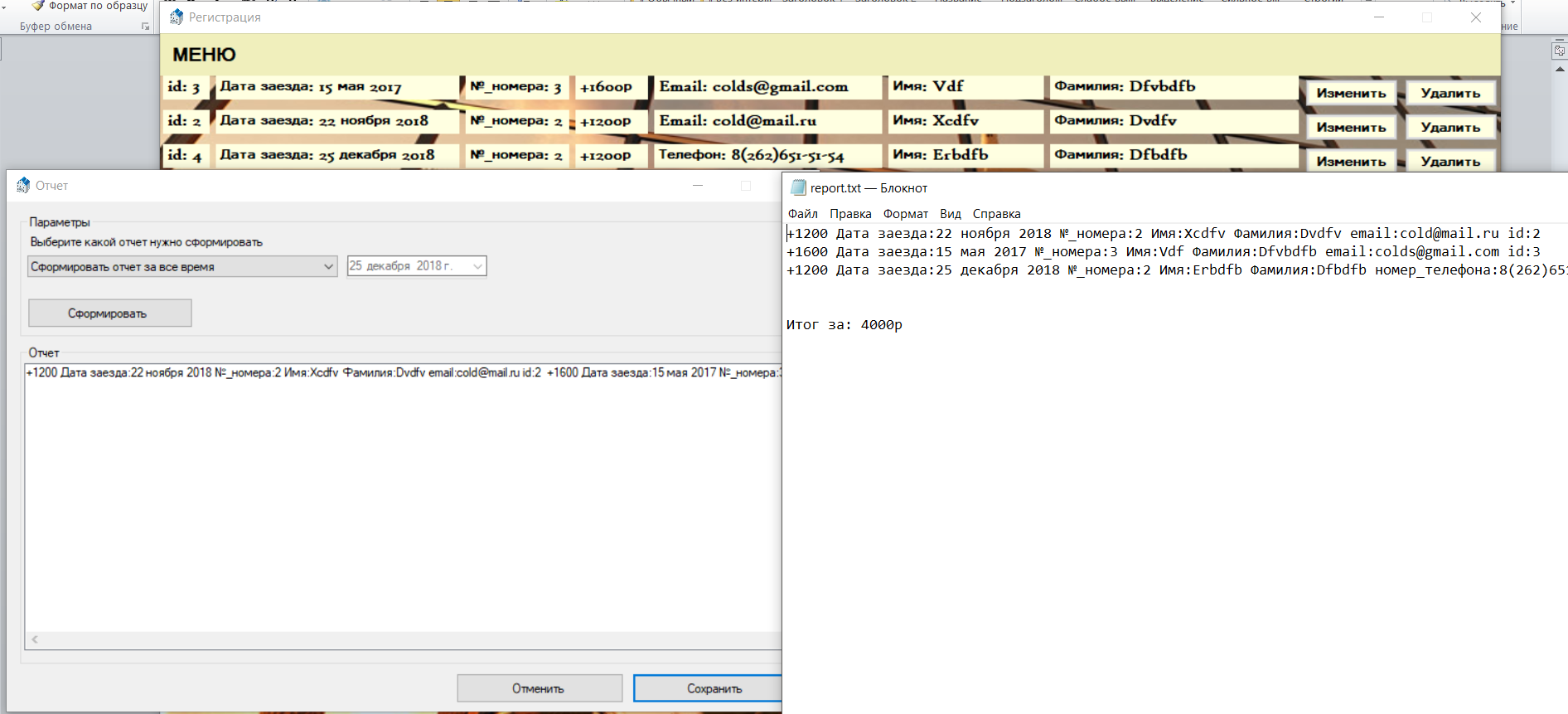


Рис. 10 – Создание отчёта и его вид в .txt файле за все время

На рисунке 11 был создан и выведен отчёт за отчетный период с заданной датой 25 декабря 2018. Сам отчёт сохраняется в виде .txt файла, и содержит записи за введённый отчетный период, а так же подсчет этой суммы за этот период.

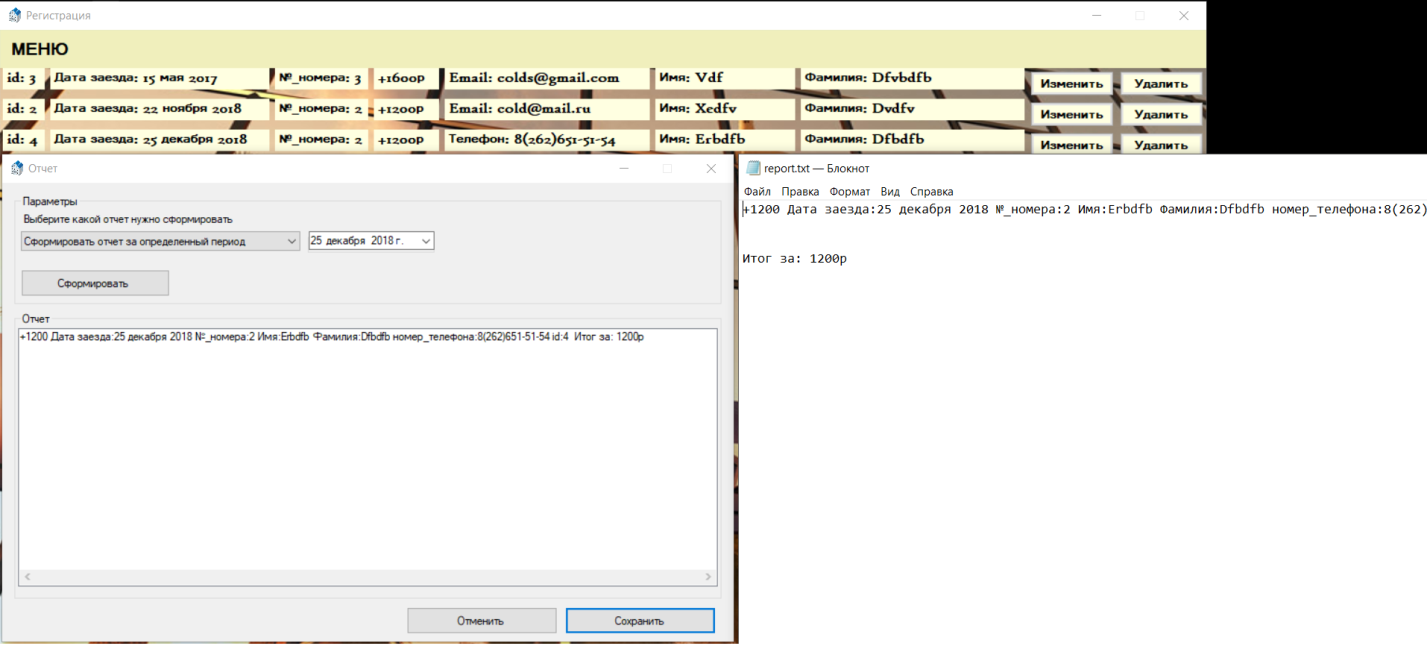


Рис. 11 – Создание отчёта и его вид в .txt файле за отчетный период