

МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт (технический университет)»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| УГНС | | 09.00.00 | Информатика и вычислительная техника | | |
| Направление подготовки | | 09.03.01 | Информатика и вычислительная техника | | |
| Направленность (профиль) | |  | Системы автоматизированного проектирования | | |
| Форма обучения | |  | очная | | |
|  | |  |  | | |
| Факультет | |  | Информационных технологий и управления | | |
| Кафедра | |  | Систем автоматизированного проектирования и управления | | |
| Учебная дисциплина | |  | Программирование | | |
| Курс | Ⅱ | | | Группа | 413 |

Отчёт по контрольной работе № 4

Вариант № 7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исполнитель: |  |  |  |  |
| обучающийся группы 413 |  |  |  | Занин Егор Валерьевич |
|  |  | (дата, подпись) |  |  |
|  |  |  |  |  |
| Проверил(и): |  |  |  | Корниенко Иван Григорьевич |
|  |  | (дата, подпись) |  | Федин Алексей Константинович |
|  |  |  |  |  |

Содержание

[1. Постановка задачи 3](#_Toc98373326)

[2. Исходные данные 3](#_Toc98373327)

[3. Особые ситуации 3](#_Toc98373328)

[4. Форматы представления данных 4](#_Toc98373330)

[5. Структура программы 5](#_Toc98373331)

[6. Блок-схемы алгоритмов программы 6](#_Toc98373332)

[7. Описание хода выполнения контрольной работы 6](#_Toc98373333)

[8. Результаты работы программы 7](#_Toc98373334)

[9. Исходный текст программы 10](#_Toc98373335)

1. Постановка задачи

Необходимо написать приложение с использованием технологии WinForms реализующие вариант задания. Программа должна позволять добавлять новые сущности с использованием интерфейса и редактировать существующие. Сущности, добавленные в программу, должны сохраняться между запусками приложения. Для хранения данных необходимо использовать СУБД SqLite. Необходимо предусмотреть возможность сохранения списка существующих сущностей в файл.

Необходимо написать программу хранения результатов искусственных астрономических объектов (искусственные спутники земли, межпланетные станции).

1. Исходные данные

В качестве исходных данных представлена предварительно заполненная данными таблица.

1. Особые ситуации
2. Если пользователь, при загрузке из файла, введет некорректное имя файла, программа будет его просить ввести корректное имя файла, до тех пор, пока он не введет корректное.
3. Если строка параметров в загружаемом файле будет неверного формата или недостаточной длины, то программа на это укажет и попросит ввести имя файла заново
4. Форматы представления данных

Программа использует следующие переменные:

Таблица 1 – Переменные, используемый в программе

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| connection | var | Соединение с БД |
| cmd | var | Запрос к БД |
| users | List<User> | Данные таблицы |
| reader | StreamReader | Заполнение таблицы данными |
| sqlexp | string | Вспомогательный запрос к БД |
| textbox | textbox | Поле для заполняемой информации |
| label | label | Поле текста |
| datagridview | datagridview | Корректное отображение данных таблицы |
| tablelayoutpanel | tablelayoutpanel | Расположение данных таблицы |
| filename | string | Имя файла |
| index | int | Индекс поля |

1. Структура программы

Программа была разделена на несколько исполняемых модулей.

Таблица 2 – Модули программы

|  |  |
| --- | --- |
| Название | Описание |
| Program | Начальное окно |
| Query | Все запросы к БД |
| User | Хранение пользовательских данных таблицы |
| Connection | Соединение с БД |
| Form1 | Начальная форма |
| Form2 | Форма добавления новой записи в таблицу БД |

Таблица 3 – Функции, составляющие модуль Program

|  |  |
| --- | --- |
| Имя | Описание |
| Main | Запуск программы |

Таблица 4 – Функции, составляющие модуль Query

|  |  |
| --- | --- |
| Имя | Описание |
| ConnectionToSQLAndShowUsers | Подключение к БД |
| AddAndShowUsers | Добавление новой записи в БД |
| UpdateAndShowUsers | Обновление данных записи БД |
| DeleteAndShowUsers | Удаление записи в таблице БД |
| TruncateAllUsers | Удаление всех записей таблицы БД |
| TakeFromFileAndShowUsers | Загрузка данных из файла |

Таблица 5 – Функции, составляющие модуль User

|  |  |
| --- | --- |
| Имя | Описание |
| DisplayName | Отображаемое имя заголовка таблицы БД |
| Browsable | Видимость заголовка столбца таблицы БД |

Таблица 6 – Функции, составляющие модуль Connection

|  |  |
| --- | --- |
| Имя | Описание |
| ConnectionToSQLAndShowUsers | Соединение с БД |

Таблица 7 – Функции, составляющие модуль Form1

|  |  |
| --- | --- |
| Имя | Описание |
| InitializeComponent | Загрузка формы |
| Button\_Click | Событие кнопки |
| dataGridView\_CellClick | Событие нажатия на ячейку таблицы |
| textBox\_KeyUp | Событие обработчика изменения текста |
| toolStripButton\_Click | Событие обработки нажатия инструмента панели инструментов |

Таблица 8 – Функции, составляющие модуль Connection

|  |  |
| --- | --- |
| Имя | Описание |
| InitializeComponent | Загрузка формы |

1. Блок-схемы алгоритмов программы

На рисунке 1 представлена блок-схема алгоритма решения задачи.



Рисунок 1 – Блок-схема решения задачи

1. Описание хода выполнения контрольной работы

При выполнении данной контрольной работой, мною было изучено: работа с базой данных SQLite.

1. Результаты работы программы

Результат тестирования всех возможностей программы приведен на рисунке 2.

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 2, лист 1 – Экранная копия результата работы разработанной программы

Изображение выглядит как текст, снимок экрана, число, диаграмма

Автоматически созданное описание

Рисунок 2, лист 2 – Экранная копия результата работы разработанной программы

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 2, лист 3 – Экранная копия результата работы разработанной программы

Изображение выглядит как текст, снимок экрана, дисплей, число

Автоматически созданное описание

Рисунок 2, лист 4 – Экранная копия результата работы разработанной программы

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 2, лист 5 – Экранная копия результата работы разработанной программы

1. Исходный текст программы

Код с файла «Program.cs»:

namespace Course\_Work4

{

internal static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

ApplicationConfiguration.Initialize();

Application.Run(new Form1());

}

}

}

Код с файла «Query.cs»:

using System.Data.SQLite;

namespace Course\_Work4

{

public class Query

{

public static List<User>? ConnectionToSQLAndShowUsers()

{

try

{

using var connection = new SQLiteConnection(@"Data Source = AstronomicalObject.db");

connection.Open();

using var cmd = new SQLiteCommand(@"select ID,

Name,

Weight,

Speed,

Material,

ServiceLife

FROM AstronomicalObjects;", connection);

using var reader = cmd.ExecuteReader();//получение строк из источника данных

List<User> users = new();

while (reader.Read())

{

users.Add(new User

{

ID = reader.GetInt32(0),

Name = reader.GetString(1),

Weight = reader.GetString(2),

Speed = reader.GetString(3),

Material = reader.GetString(4),

ServiceLife = reader.GetString(5)

});

}

return users;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

return null;

}

public static List<User>? AddAndShowUsers(string name,

string weight,

string speed,

string material,

string servicelife,

int id)

{

try

{

string sqlexp1 = $"UPDATE sqlite\_sequence SET seq={id} WHERE name = 'AstronomicalObjects'";

string sqlexp = @"insert into AstronomicalObjects(

Name,

Weight,

Speed,

Material,

ServiceLife)

values(@name, @weight, @speed, @material, @servicelife)";

using var connection = new SQLiteConnection(@"Data Source = AstronomicalObject.db");

connection.Open();

var cm1 = new SQLiteCommand(sqlexp1, connection);

var cmd = new SQLiteCommand(sqlexp, connection);

SQLiteParameter Name = new("@name", name);

cmd.Parameters.Add(Name);

SQLiteParameter Weight = new("@weight", weight);

cmd.Parameters.Add(Weight);

SQLiteParameter Speed = new("@speed", speed);

cmd.Parameters.Add(Speed);

SQLiteParameter Material = new("@material", material);

cmd.Parameters.Add(Material);

SQLiteParameter ServiceLife = new("@servicelife", servicelife);

cmd.Parameters.Add(ServiceLife);

cm1.ExecuteNonQuery();

cmd.ExecuteNonQuery();

cmd.CommandText = "select \* from AstronomicalObjects";

using var reader = cmd.ExecuteReader();

List<User> users = new();

while (reader.Read())

{

users.Add(new User

{

ID = reader.GetInt32(0),

Name = reader.GetString(1),

Weight = reader.GetString(2),

Speed = reader.GetString(3),

Material = reader.GetString(4),

ServiceLife = reader.GetString(5)

});

}

return users;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

return null;

}

public static List<User>? UpdateAndShowUsers(string name,

string weight,

string speed,

string material,

string servicelife,

int id)

{

try

{

string sqlexp = $@"UPDATE AstronomicalObjects

set Name = @name,

Weight = @weight,

Speed = @speed,

Material = @material,

ServiceLife = @servicelife

where ID = @id";

using var connection = new SQLiteConnection(@"Data Source = AstronomicalObject.db");

connection.Open();

var cmd = new SQLiteCommand(sqlexp, connection);

SQLiteParameter Name = new("@name", name);

cmd.Parameters.Add(Name);

SQLiteParameter Weight = new("@weight", weight);

cmd.Parameters.Add(Weight);

SQLiteParameter Speed = new("@speed", speed);

cmd.Parameters.Add(Speed);

SQLiteParameter Material = new("@material", material);

cmd.Parameters.Add(Material);

SQLiteParameter ServiceLife = new("@servicelife", servicelife);

cmd.Parameters.Add(ServiceLife);

SQLiteParameter ID = new("@id", id);

cmd.Parameters.Add(ID);

cmd.ExecuteNonQuery();

cmd.CommandText = "select \* from AstronomicalObjects";

using var reader = cmd.ExecuteReader();

List<User> users = new();

while (reader.Read())

{

users.Add(new User

{

ID = reader.GetInt32(0),

Name = reader.GetString(1),

Weight = reader.GetString(2),

Speed = reader.GetString(3),

Material = reader.GetString(4),

ServiceLife = reader.GetString(5)

});

}

return users;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

return null;

}

public static List<User>? DeleteAndShowUsers(int id)

{

try

{

string sqlexp = $@"delete from AstronomicalObjects where ID = {id}";

using var connection = new SQLiteConnection(@"Data Source = AstronomicalObject.db");

connection.Open();

var cmd = new SQLiteCommand(sqlexp, connection);

cmd.ExecuteNonQuery();

cmd.CommandText = "select \* from AstronomicalObjects";

using var reader = cmd.ExecuteReader();

List<User> users = new();

while (reader.Read())

{

users.Add(new User

{

ID = reader.GetInt32(0),

Name = reader.GetString(1),

Weight = reader.GetString(2),

Speed = reader.GetString(3),

Material = reader.GetString(4),

ServiceLife = reader.GetString(5)

});

}

return users;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

return null;

}

public static List<User>? TruncateAllUsers()

{

try

{

string sqlexp = $@"delete from AstronomicalObjects";

using var connection = new SQLiteConnection(@"Data Source = AstronomicalObject.db");

connection.Open();

var cmd = new SQLiteCommand(sqlexp, connection);

cmd.ExecuteNonQuery();

cmd.CommandText = "select \* from AstronomicalObjects";

using var reader = cmd.ExecuteReader();

List<User> users = new();

return users;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

return null;

}

public static List<User>? TakeFromFileAndShowUsers(string name,

string weight,

string speed,

string material,

string servicelife)

{

try

{

string sqlexp1 = "UPDATE sqlite\_sequence SET seq=0 WHERE name = 'AstronomicalObjects'";

string sqlexp = @"insert into AstronomicalObjects(

Name,

Weight,

Speed,

Material,

ServiceLife)

values(@name, @weight, @speed, @material, @servicelife)";

using var connection = new SQLiteConnection(@"Data Source = AstronomicalObject.db");

connection.Open();

var cm1 = new SQLiteCommand(sqlexp1, connection);

var cmd = new SQLiteCommand(sqlexp, connection);

SQLiteParameter Name = new("@name", name);

cmd.Parameters.Add(Name);

SQLiteParameter Weight = new("@weight", weight);

cmd.Parameters.Add(Weight);

SQLiteParameter Speed = new("@speed", speed);

cmd.Parameters.Add(Speed);

SQLiteParameter Material = new("@material", material);

cmd.Parameters.Add(Material);

SQLiteParameter ServiceLife = new("@servicelife", servicelife);

cmd.Parameters.Add(ServiceLife);

cm1.ExecuteNonQuery();

cmd.ExecuteNonQuery();

cmd.CommandText = "select \* from AstronomicalObjects";

using var reader = cmd.ExecuteReader();

List<User> users = new();

while (reader.Read())

{

users.Add(new User

{

ID = reader.GetInt32(0),

Name = reader.GetString(1),

Weight = reader.GetString(2),

Speed = reader.GetString(3),

Material = reader.GetString(4),

ServiceLife = reader.GetString(5)

});

}

return users;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

return null;

}

}

}

Код с файла «User.cs»:

using System.ComponentModel;

namespace Course\_Work4

{

public class User

{

[DisplayName("ID")]

public int ID { get; internal set; }

[DisplayName("Название")]

public string? Name { get; internal set; }

[DisplayName("Масса(тонн)")]

public string? Weight { get; internal set; }

[DisplayName("Скорость(км/ч)")]

public string? Speed { get; internal set; }

[DisplayName("Материал")]

public string? Material { get; internal set; }

[DisplayName("Срок службы(лет)")]

public string? ServiceLife { get; internal set; }

}

}

Код с файла «Form1.cs»:

using System;

using System.Data.Entity.Core.Mapping;

using System.Data.SQLite;

using System.Windows.Forms;

using static Course\_Work4.Query;

namespace Course\_Work4

{

public partial class Form1 : Form

{

private readonly List<User> \_list;

private string? checkText;

public Form1()

{

InitializeComponent();

using (StreamReader reader = new("MenuStatus.txt"))

{

string line;

line = reader.ReadLine()!;

if (Convert.ToInt32(line) == 1)

{

MessageBox.Show("Практическая работа №4.\r\n " +

"Многопроцессорное и многопоточное программирование\r\n " +

"Хранение искусственных астрономических объектов.\r\n " +

"Студент группы 413, Занин Егор Валерьевич. 2023 год",

"Сообщение",

MessageBoxButtons.OK,

MessageBoxIcon.Information,

MessageBoxDefaultButton.Button1,

MessageBoxOptions.DefaultDesktopOnly);

ShowMenu.Checked = true;

}

}

\_list = new List<User>();

bsUsers.DataSource = \_list;

List<User>? list = ConnectionToSQLAndShowUsers();

if (list != null && list.Count > 0)

{

\_list.AddRange(list);

bsUsers.ResetBindings(false);

}

dataGridView1.AutoGenerateColumns = true;

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox2.Visible == true && CheckUpdateData())

{

int row = dataGridView1.CurrentCell.RowIndex;

int index = (int)dataGridView1.Rows[row].Cells[0].Value;

\_list.Clear();

List<User>? list = UpdateAndShowUsers(comboBox1.Text,

textBox2.Text,

textBox3.Text,

textBox4.Text,

textBox5.Text,

index);

if (list != null && list.Count > 0)

{

\_list.AddRange(list);

bsUsers.ResetBindings(false);

}

}

\_list.Clear();

List<User>? list1 = ConnectionToSQLAndShowUsers();

if (list1 != null && list1.Count > 0)

{

\_list.AddRange(list1);

bsUsers.ResetBindings(false);

}

}

private bool CheckUpdateData()

{

try

{

if (comboBox1.Text != "" && textBox4.Text != "")

{

textBox2.Text = Convert.ToDouble(textBox2.Text).ToString();

textBox3.Text = Convert.ToDouble(textBox3.Text).ToString();

textBox5.Text = Convert.ToDouble(textBox5.Text).ToString();

}

else

{

MessageBox.Show(this, "Неверный формат данных.\n" +

"Измените данные и повторите ввод", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

return false;

}

}

catch

{

MessageBox.Show(this, "Неверный формат данных.\n" +

"Измените данные и повторите ввод", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

return false;

}

return true;

}

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

{

Form form2 = new Form2();

form2.Show();

Visible = false;

}

public void FillDatabase()

{

Form2? form2 = Application.OpenForms.OfType<Form2>().FirstOrDefault(); // Поиск уже созданного экземпляра form2

if (form2 != null)

{

int row = dataGridView1.CurrentCell.RowIndex;

int index = (int)dataGridView1.Rows[row].Cells[0].Value;

\_list.Clear();

List<User>? list = AddAndShowUsers(form2.comboBox1.Text,

form2.textBox2.Text,

form2.textBox3.Text,

form2.textBox4.Text,

form2.textBox5.Text,

index);

if (list != null && list.Count > 0)

{

\_list.AddRange(list);

bsUsers.ResetBindings(false);

}

}

}

private void button3\_Click(object sender, EventArgs e)

{

int row = dataGridView1.CurrentCell.RowIndex;

int index = (int)dataGridView1.Rows[row].Cells[0].Value;

\_list.Clear();

DialogResult result = MessageBox.Show("Вы точно хотите удалить запись?",

"Сообщение",

MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

List<User>? list = DeleteAndShowUsers(index);

if (list != null && list.Count > 0)

{

\_list.AddRange(list);

bsUsers.ResetBindings(false);

}

}

else

{

List<User>? list = ConnectionToSQLAndShowUsers();

if (list != null && list.Count > 0)

{

\_list.AddRange(list);

bsUsers.ResetBindings(false);//обновить значения

}

}

}

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

comboBox1.Visible = true;

textBox2.Visible = true;

textBox3.Visible = true;

textBox4.Visible = true;

textBox5.Visible = true;

label1.Visible = true;

label2.Visible = true;

label3.Visible = true;

label4.Visible = true;

label5.Visible = true;

checkText = dataGridView1.CurrentRow.Cells[1].Value.ToString();

if (checkText == "Искусственный спутник Земли")

{

comboBox1.SelectedIndex = 0;

}

else

{

comboBox1.SelectedIndex = 1;

}

textBox2.Text = dataGridView1.CurrentRow.Cells[2].Value.ToString();

textBox3.Text = dataGridView1.CurrentRow.Cells[3].Value.ToString();

textBox4.Text = dataGridView1.CurrentRow.Cells[4].Value.ToString();

textBox5.Text = dataGridView1.CurrentRow.Cells[5].Value.ToString();

}

private void toolStripButton1\_Click(object sender, EventArgs e)

{

if (saveFileDialog1.ShowDialog() == DialogResult.Cancel)

return;

string filename = saveFileDialog1.FileName;

using StreamWriter writer = new(filename, false);

for (int i = 0; i < dataGridView1.RowCount; i++)

{

for (int j = 1; j < dataGridView1.ColumnCount; j++)

{

string? text = dataGridView1.Rows[i].Cells[j].Value.ToString();

writer.WriteLine(text);

}

}

}

private void toolStripButton2\_Click(object sender, EventArgs e)

{

if (openFileDialog1.ShowDialog() == DialogResult.Cancel)

return;

string filename = openFileDialog1.FileName;

using StreamReader reader = new(filename);

try

{

string[] dataFromFile = File.ReadAllLines(filename);

if (dataFromFile.Length % 5 != 0)

return;

\_list.Clear();

List<User>? list = TruncateAllUsers();

string[,] dataFromTheFile = new string[dataFromFile.Length / 5, dataGridView1.ColumnCount];

int counter = 0;

for (int i = 0; i < (dataFromFile.Length / 5); i++)

{

for (int j = 0; j < 5; j++)

{

dataFromTheFile[i, j] = dataFromFile[counter];

counter++;

}

list = TakeFromFileAndShowUsers(dataFromTheFile[i, 0],

dataFromTheFile[i, 1],

dataFromTheFile[i, 2],

dataFromTheFile[i, 3],

dataFromTheFile[i, 4]);

}

if (list != null && list.Count > 0)

{

\_list.AddRange(list);

bsUsers.ResetBindings(false);

}

}

catch

{

MessageBox.Show(this, "Произошла ошибка при попытке считывания данных из файла." +

"Отредактируйте данные и попробуйте снова", "Ошибка!",

MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

using StreamWriter writer = new("MenuStatus.txt", false);

if (ShowMenu.Checked == true)

{

ShowMenu.Checked = false;

}

else

{

ShowMenu.Checked = true;

}

if (ShowMenu.Checked == true)

{

writer.WriteLine(1);

}

else

{

writer.WriteLine(0);

}

}

private void textBox2\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8 && number != 44) // цифры, клавиша BackSpace и запятая

{

e.Handled = true;

}

}

private void textBox3\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8 && number != 44) // цифры, клавиша BackSpace и запятая

{

e.Handled = true;

}

}

private void textBox5\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8) // цифры, клавиша BackSpace

{

e.Handled = true;

}

}

}

}

Код с файла «Form2.cs»:

namespace Course\_Work4

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

private void textBox2\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8 && number != 44) // цифры, клавиша BackSpace и запятая

{

e.Handled = true;

}

}

private void textBox3\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8 && number != 44) // цифры, клавиша BackSpace и запятая

{

e.Handled = true;

}

}

private void textBox5\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8) // цифры, клавиша BackSpace

{

e.Handled = true;

}

}

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

{

try

{

if (comboBox1.Text != "" && textBox4.Text != "")

{

textBox2.Text = Convert.ToDouble(textBox2.Text).ToString();

textBox3.Text = Convert.ToDouble(textBox3.Text).ToString();

textBox5.Text = Convert.ToDouble(textBox5.Text).ToString();

Form1? form1 = Application.OpenForms.OfType<Form1>().FirstOrDefault(); // Поиск уже созданного экземпляра form2

if (form1 != null)

{

form1.FillDatabase();

form1.Visible = true;

}

Close();

}

else

{

MessageBox.Show(this, "Неверный формат данных.\n" +

"Измените данные и повторите ввод", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

catch

{

MessageBox.Show(this, "Неверный формат данных.\n" +

"Измените данные и повторите ввод", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void button1\_Click(object sender, EventArgs e)

{

Form1? form1 = Application.OpenForms.OfType<Form1>().FirstOrDefault(); // Поиск уже созданного экземпляра form2

if (form1 != null)

{

form1.Visible = true;

}

Close();

}

}

}