Министерство образования Российской Федерации Пензенский государственный университет

Кафедра «Математическое обеспечение и применение ЭВМ»

**ОТЧЕТ**

по лабораторной работе №1

по курсу «ООП» на тему «Класс. Статические члены класса. Обработка исключений.»

Выполнил:

студент группы 23ВП1

Рейимов М.

Принял:

к.т.н. Афонин А.Ю.

Пенза 2025

### Название

Класс. Статические члены класса. Обработка исключений.

### Цель работы

### Реализовать класс. Создать на основе класса объект, вывести и изменить содержимое его полей, запросив новые значения от пользователя. Научиться создавать приложения с графическим интерфейсом пользователя (GUI) на основе Windows Forms. Подробно ознакомиться с конструкторами объектов. Научиться использовать статические члены классов. Научиться обрабатывать исключительные ситуации в процессе работы программы. Вызов неуправляемого кода.

### Лабораторное задание

Реализовать класс, содержащий не менее 7 полей с различными типами данных:

• cтрока;

• целое;

• вещественное.

Класс должен быть оформлен в виде отдельного файла. Создать на основе класса объект. Класс должен содержать конструктор без параметров, с одним параметром, с двумя параметрами, с количеством параметров, равным количеству полей класса. Вывести содержимое его полей:

• с использование переопределенного метода ToString() для класса в целом;

• непосредственно обращаясь к полю;

• реализовав метод, выводящий на экран значение определенного поля;

• одно из числовых полей вывести в шестнадцатеричном представлении.

Поля должны быть инициализированы посредством передачи параметров конструктору. Добавить возможность изменения полей каждого из типов данных, запросив их значения от пользователя.

Работа приложения должна начинаться с вывода приветственной строки с названием лабораторной работы, группы и составом бригады.

Реализовать в классе счетчик созданных объектов, задействовать статический член класса. Изменение и вывод информации в объектах класса организовать через GUI. Вывести содержимое класса на экран пользователю.

Реализовать в коде лабораторной работы перехват и обработку исключений, при этом переопределив с помощью наследования исключение (номер варианта делим по модулю 7):

1) StackOverflowException

2) ArrayTypeMismatchException

3) DivideByZeroException

4) IndexOutOfRangeException

5) InvalidCastException

6) OutOfMemoryException

7) OverflowException

Дополнить исключение своим блоком данных. Заполнить его в момент генерации. Отобразить в момент перехвата.

Реализовать свой тип исключения. Сгенерировать и выполнить его перехват.

На возникновение исключения отреагировать выводом системного окна сообщения посредством вызова неуправляемого кода: функции MessageBox из библиотеки user32.dll (Не используем класс .Net MessageBox).

### Предметная область – фирма грузоперевозок.

### Листинг

### Файл TransportCompany.cs:

using System;

using System.Collections.Generic;

using System.Drawing.Printing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab1

{

public class TransportCompany

{

public string name { get; set; }

public int pricePerKilometer { get; set; }

public float averageDeliveryTime { get; set; }

public int yearFounded { get; set; }

public float transportedMass { get; set; }

public float rating { get; set; }

public string phoneNumber { get; set; }

public static int countObj = 0;

public TransportCompany()

{

name = "";

pricePerKilometer = 0;

averageDeliveryTime = 0;

yearFounded = 2000;

transportedMass = 0;

rating = 0;

phoneNumber = "";

countObj++;

}

public TransportCompany(string name) : this()

{

this.name = name;

}

public TransportCompany(string name, int pricePerKilometer) : this(name)

{

this.pricePerKilometer = pricePerKilometer;

}

public TransportCompany(string name, int pricePerKilometer, float averageDeliveryTime, int yearFounded, float transportedMass, float rating, string phoneNumber) : this(name, pricePerKilometer)

{

this.averageDeliveryTime = averageDeliveryTime;

this.yearFounded = yearFounded;

this.transportedMass = transportedMass;

this.rating = rating;

this.phoneNumber = phoneNumber;

}

public override string ToString()

{

return "Название фирмы: " + name +

"\nЦена за километр: " + pricePerKilometer +

"\nСреднее время доставки: " + averageDeliveryTime +

"\nГод основания: " + yearFounded +

"\nМасса перевезенных грузов " + transportedMass +

"\nРейтинг: " + rating +

"\nНомер телефона: " + phoneNumber;

}

public string PrintName()

{

if (string.IsNullOrEmpty(name))

return "У фирмы нет названия";

return "Название фирмы: " + name;

}

public string PriceToHex()

{

return Convert.ToString(pricePerKilometer, 16);

}

}

}

Файл MyException.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab1

{

internal class MyException : Exception

{

public MyException(string message) : base(message) { }

}

}

Файл Form1.cs:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Security.Policy;

using System.Text;

using System.Threading.Tasks;

using System.Text.RegularExpressions;

using System.Windows.Forms;

namespace Lab1

{

public partial class Form1 : Form

{

TransportCompany firm;

public Form1()

{

InitializeComponent();

MessageBox.Show("Класс. Статические члены класса. Обработка исключений.\n" +

"Рейимов М. 23ВП1\nВариант 8. Фирма грузоперевозок", "Лабораторная работа 1");

}

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

{

try

{

if (string.IsNullOrWhiteSpace(nameInput.Text))

throw new MyException("Фирма должна иметь название");

if (string.IsNullOrWhiteSpace(phonenumberInput.Text))

throw new MyException("Фирма должна иметь номер");

if (!Regex.IsMatch(phonenumberInput.Text.Trim(), @"^\d{11}$"))

throw new MyException("Номер должен состоять из 11 цифр и не содержать буквы или символы");

firm = new TransportCompany(nameInput.Text,

(int)priceInput.Value,

(float)avgtimeInput.Value,

(int)yearInput.Value,

(float)massInput.Value,

(float)ratingInput.Value,

phonenumberInput.Text);

info.Text = firm.ToString();

dataGridView1.Rows.Clear();

dataGridView1.RowCount = 1;

dataGridView1.Rows[0].Cells[0].Value = nameInput.Text;

dataGridView1.Rows[0].Cells[1].Value = priceInput.Value;

dataGridView1.Rows[0].Cells[2].Value = avgtimeInput.Value;

dataGridView1.Rows[0].Cells[3].Value = yearInput.Value;

dataGridView1.Rows[0].Cells[4].Value = massInput.Value;

dataGridView1.Rows[0].Cells[5].Value = ratingInput.Value;

dataGridView1.Rows[0].Cells[6].Value = phonenumberInput.Text;

objCount.Text = TransportCompany.countObj.ToString();

}

catch (MyException ex)

{

MessageBox.Show(ex.Message, "Ошибка");

}

}

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

{

try

{

if (firm == null)

throw new MyException("Объект класса не создан");

info.Text = firm.PrintName();

}

catch (MyException ex)

{

MessageBox.Show(ex.Message, "Ошибка");

}

}

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

{

try

{

if (firm == null)

throw new MyException("Объект класса не создан");

info.Text = "Цена за километр\nв шестнадцатиричном представлении: " + firm.PriceToHex();

}

catch (MyException ex)

{

MessageBox.Show(ex.Message, "Ошибка");

}

}

}

}

### Пояснительный текст к программе

Класс TransportCompany представляет собой модель транспортной компании и содержит поля name (наименование компании), pricePerKilometer (цена за километр), averageDeliveryTime (среднее время доставки), yearFounded (год основания), transportedMass (масса перевезенного груза), rating (рейтинг компании), phoneNumber (номер телефона).

Класс MyException – это класс пользовательских исключений. Является наследником базового класса Exception и содержит информацию об ошибке, сгенерированной во время выполнения программы.

### Результаты работы программы

Результаты работы программы показаны на рисунках 1-4.

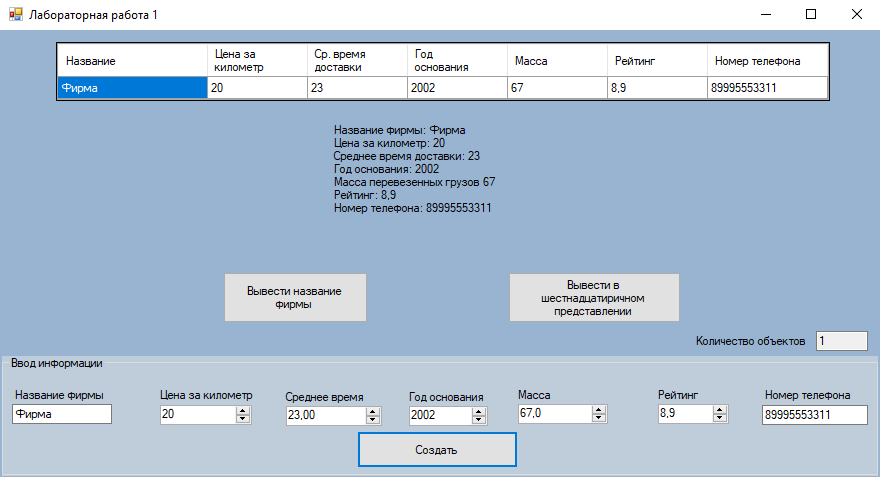


Рисунок 1 – Создание объекта и вывод его полей

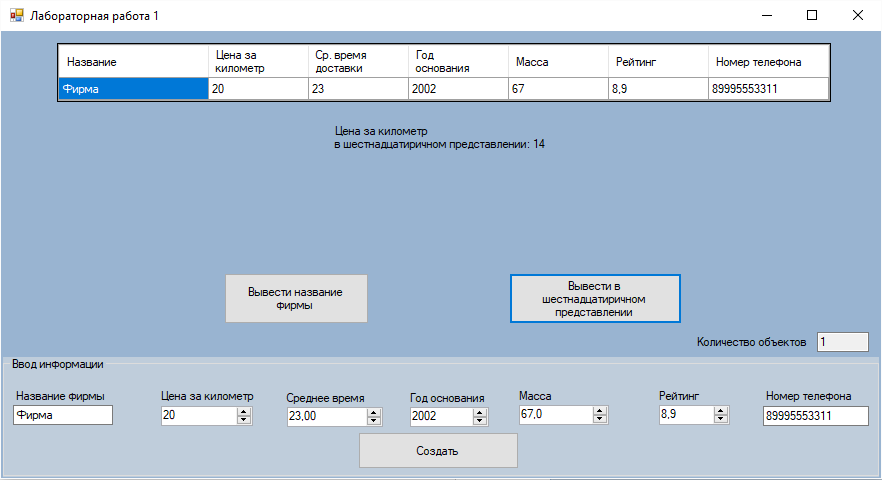


Рисунок 2 – Шестнадцатеричное представление поля «цена за километр»

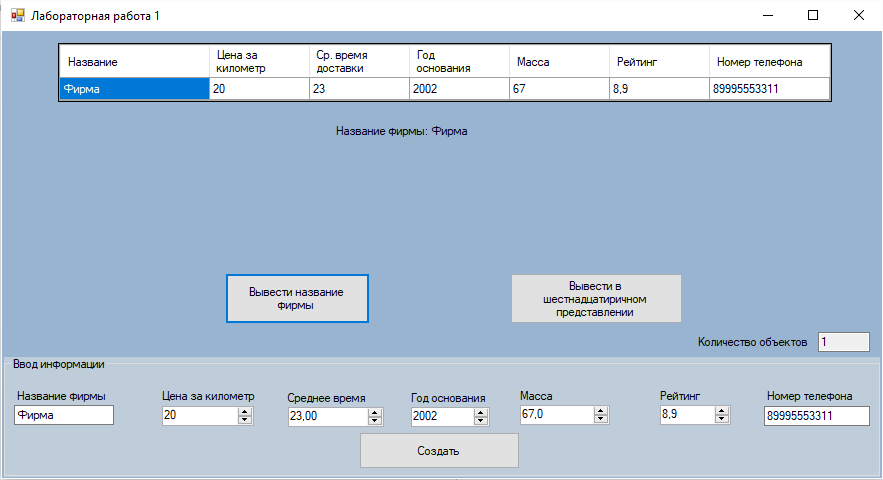


Рисунок 3 – Вывод поля «наименование фирмы»

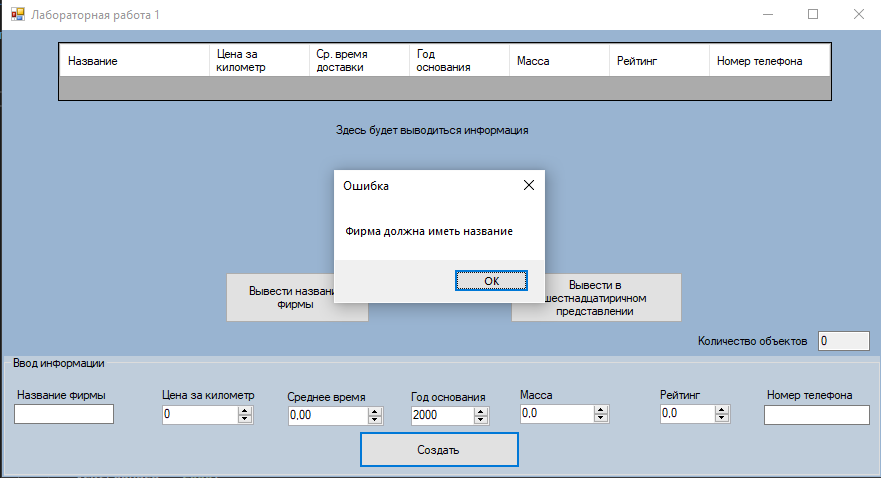


Рисунок 4 – Обработка пользовательского исключения

### Диаграмма вариантов использования

### 

Рисунок 5 - Диаграмма вариантов использования

### Выводы

В ходе выполнения лабораторной работы были выполнены следующие задачи. Был разработан класс TransportCompany, создан его объект, а также проведены операции по выводу и изменению значений его полей. Это позволило закрепить навыки работы с классами, объектами и их свойствами. Освоены основы разработки приложений с графическим интерфейсом пользователя (GUI) на платформе Windows Forms. Были изучены базовые элементы управления и их взаимодействие с кодом. Ознакомились с принципами создания и использования конструкторов объектов, что позволило лучше понять механизмы инициализации объектов в C#. Был создан пользовательский класс исключений MyException, а также реализованы механизмы генерации и обработки исключений этого класса. Перехват и обработка исключения StackOverflowException не были реализованы, так как задание не предусматривало ситуаций, приводящих к переполнению стека. В связи с этим данное исключение не могло быть сгенерировано в рамках текущей работы.