Министерство образования Российской Федерации Пензенский государственный университет

Кафедра «Математическое обеспечение и применение ЭВМ»

**ОТЧЕТ**

по лабораторной работе №4

по курсу «ООП» на тему «Структурные паттерны»

Выполнил:

студент группы 23ВП1 Рейимов М.

Принял:

к.т.н. Афонин А.Ю.

Пенза 2025

### Название

Структурные паттерны.

### Цель работы

Научиться использовать структурные паттерны.

### Лабораторное задание

Реализовать структурный паттерн программирования «мост».

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 interface IDeliveryType

{

string GetDeliveryType();

}

public class SeaDelivery : IDeliveryType

{

public string GetDeliveryType()

{

return "По воде";

}

}

public class RoadDelivery : IDeliveryType

{

public string GetDeliveryType()

{

return "По земле";

}

}

public class AirDelivery : IDeliveryType

{

public string GetDeliveryType()

{

return "По воздуху";

}

}

public abstract 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 IDeliveryType deliveryType;

public static int countObj = 0;

public TransportCompany()

{

name = "";

pricePerKilometer = 0;

averageDeliveryTime = 0;

yearFounded = 2000;

transportedMass = 0;

rating = 0;

phoneNumber = "";

countObj++;

deliveryType = null;

}

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, IDeliveryType deliveryType) : this(name, pricePerKilometer)

{

this.averageDeliveryTime = averageDeliveryTime;

this.yearFounded = yearFounded;

this.transportedMass = transportedMass;

this.rating = rating;

this.phoneNumber = phoneNumber;

this.deliveryType = deliveryType;

}

public virtual string GetDeliveryInfo()

{

return deliveryType.GetDeliveryType();

}

}

public class ExpressCompany : TransportCompany

{

public ExpressCompany(string name,

int pricePerKilometer,

float averageDeliveryTime,

int yearFounded,

float transportedMass,

float rating,

string phoneNumber,

IDeliveryType deliveryType) : base(name, pricePerKilometer, averageDeliveryTime, yearFounded, transportedMass, rating, phoneNumber, deliveryType) { }

public override string GetDeliveryInfo()

{

return base.GetDeliveryInfo();

}

}

}

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 StackTransportCompany.cs:

using Lab1;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab2

{

public class StackTransportCompany

{

private Stack<TransportCompany> transportCompanies;

public delegate void AddToStack(TransportCompany company);

public delegate void RemoveFromStack();

public event AddToStack StackAdded;

public event RemoveFromStack StackRemoved;

public StackTransportCompany()

{

this.transportCompanies = new Stack<TransportCompany>();

}

public Stack<TransportCompany> GetTransportCompanies()

{

return transportCompanies;

}

public void AddCompany(TransportCompany company)

{

transportCompanies.Push(company);

StackAdded?.Invoke(company);

}

public void DeleteCompany()

{

if (transportCompanies.Count == 0)

throw new MyException("Стек пуст");

TransportCompany top = transportCompanies.Pop();

StackRemoved?.Invoke();

}

}

}

Файл StackListener.cs:

using Lab1;

using System;

using System.Windows.Forms;

namespace Lab2

{

internal class StackListener

{

private DataGridView dataGridView;

private TextBox objCount;

public StackListener(StackTransportCompany stack, DataGridView dataGridView, TextBox objCount)

{

this.dataGridView = dataGridView;

this.objCount = objCount;

stack.StackAdded += (TransportCompany company) =>

{

int rowIndex = dataGridView.Rows.Add();

dataGridView.Rows[rowIndex].Cells[0].Value = company.name;

dataGridView.Rows[rowIndex].Cells[1].Value = company.pricePerKilometer;

dataGridView.Rows[rowIndex].Cells[2].Value = company.averageDeliveryTime;

dataGridView.Rows[rowIndex].Cells[3].Value = company.yearFounded;

dataGridView.Rows[rowIndex].Cells[4].Value = company.transportedMass;

dataGridView.Rows[rowIndex].Cells[5].Value = company.rating;

dataGridView.Rows[rowIndex].Cells[6].Value = company.phoneNumber;

dataGridView.Rows[rowIndex].Cells[7].Value = company.GetDeliveryInfo();

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

};

stack.StackRemoved += () =>

{

if (dataGridView.RowCount > 0)

dataGridView.Rows.RemoveAt(dataGridView.RowCount - 1);

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

};

}

}

}

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;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

using Lab1;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.ListView;

using System.Diagnostics;

using System.Xml.Linq;

namespace Lab2

{

public partial class Form1 : Form

{

private StackTransportCompany companies;

private StackListener stackListener;

public Form1()

{

InitializeComponent();

MessageBox.Show("Структурные паттерны. Мост.\n" +

"Рейимов М. 23ВП1\nВариант 8. Фирма грузоперевозок", "Лабораторная работа 4");

companies = new StackTransportCompany();

stackListener = new StackListener(companies, dataGridView1, objCount);

}

private void create\_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");

if (deliveryMethod.SelectedIndex == -1)

throw new MyException("Выберите метод доставки");

string type = deliveryMethod.SelectedItem.ToString();

IDeliveryType delivery = null;

switch (type)

{

case "По воздуху":

delivery = new AirDelivery();

break;

case "По воде":

delivery = new SeaDelivery();

break;

case "По земле":

delivery = new RoadDelivery();

break;

}

TransportCompany firm = new ExpressCompany(nameInput.Text,

(int)priceInput.Value,

(float)avgtimeInput.Value,

(int)yearInput.Value,

(float)massInput.Value,

(float)ratingInput.Value,

phonenumberInput.Text,

delivery);

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

companies.AddCompany(firm);

}

catch (MyException ex)

{

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

}

}

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

{

try

{

TransportCompany.countObj--;

companies.DeleteCompany();

}

catch (MyException ex)

{

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

}

}

}

}

### Пояснительный текст к программе

Абстрактный класс TransportCompany представляет общие характеристики всех транспортных компаний 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), deliveryType (способ доставки).

Интерфейс IDeliveryType определяет общий контракт для всех способов доставки. У него есть единственный метод GetDeliveryType().

В коде реализованы три класса, наследующие этот интерфейс:

* SeaDelivery – доставка по воде.
* RoadDelivery – доставка по земле.
* AirDelivery – доставка по воздуху.

Класс ExpressCompany - это конкретная реализация транспортной компании, которая может использовать один из способов доставки.

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

Класс StackTransportCompany является классом, хранящим стек из транспортных компаний. Осуществляет добавление элемента в стек и удаление элемента из стека.

Класс StackListener – класс-слушатель, который обрабатывает события.

### Результаты работы программы

Результаты работы программы показаны на рисунках 1-4.

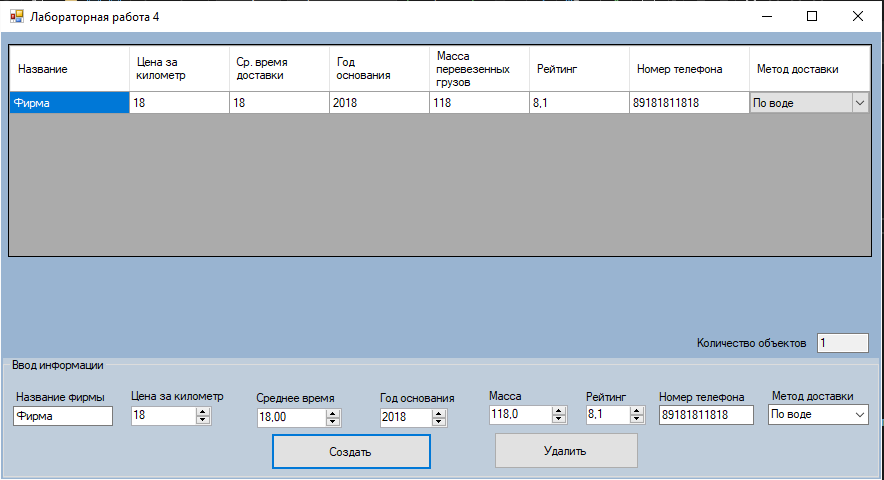


Рисунок 1 – Создание объекта, добавление в стек и вывод информации

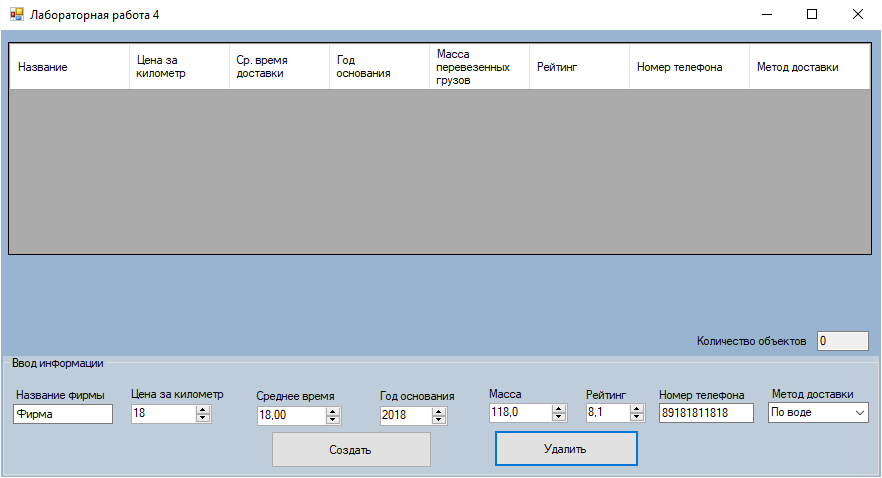


Рисунок 2 – Удаление фирмы из стека и вывод информации

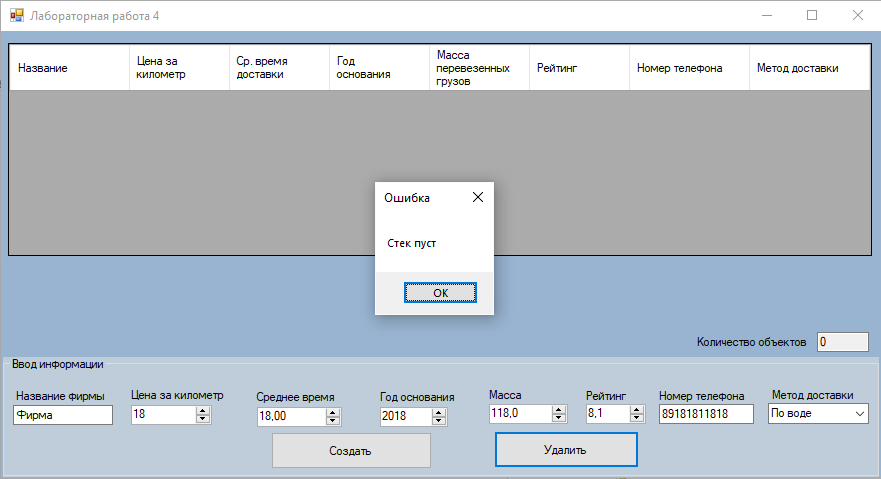


Рисунок 3 – Попытка удаления элемента из пустого стека

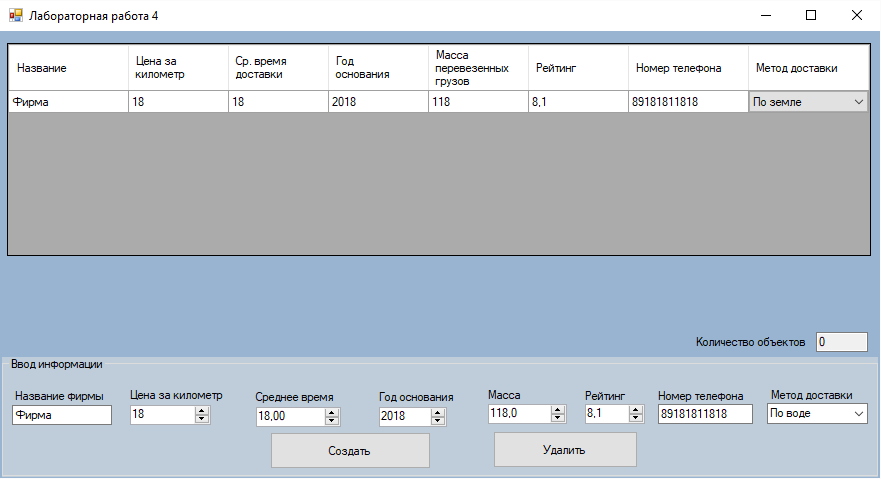
**

Рисунок 4 – Попытка изменения способа перевозки и сохранения изменений

### Диаграмма вариантов использования

### C:\Users\Влад\Downloads\2025-05-10_17-40-48.png

Рисунок 5 - Диаграмма вариантов использования

### Выводы

В ходе выполнения лабораторной работы был реализован структурный паттерн «Мост». «Мост» - структурный шаблон проектирования, который позволяет отделить абстракцию от реализации таким образом, чтобы и абстракцию, и реализацию можно было изменять независимо друг от друга. В данной лабораторной работе этот паттерн позволил независимо от абстракции изменять способ перевозки путем изменения поля объекта.