Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по программе «Задание 6 - Индивидуальное задание»

Выполнил: Меньшиков Артур Андреевич

Группа: ПР-32

Преподаватель: Мирошниченко Г. В.

2025

**Задание:**

Задание 6

Используя методы Linq и коллекции (использовать не менее 2-х видов

коллекций)

Описать базовый класс с указанными в задании полями и методами:

* Конструктор;
* Добавить 2 своих поля;
* Функция, которая определяет качество объекта – Q по заданной

формуле;

* Вывод информации об объекте.
* Реализуйте и проверьте методы для добавления и удаления

объекта базового класса (сделать 3 перегрузки!);

Построить класс потомок, который содержит:

* Дополнительное поле Р;
* Добавить еще 3 свойства
* Функция, которая определяет «качество» объекта класса потомка

– Qp, которая перекрывает функцию качества базового класса,

выполняя вычисления по новой формуле.

* Добавить свою функцию для класса потомка

Создать проект для демонстрации работы: ввод-вывод информации об

объектах базового класса и класса потомка.

Разработать модульные тесты к реализованным методам классов

2.

Дорожные работы:

- ширина дороги (м)

- длина (м)

- масса дорожного покрытия на 1 кв.м

(кг)

- Q=ширина \* длину \*масса/1000

P: коэффициент прочности в

зависимости от погодных условий

Qp= Q\*1,1 (если P = от 5 до 8)

Если P=3, 4, 9, 10 то Qp=Q\*1,6

иначе Qp=1,9\*Q

**Входные и выходные данные**

* txtWidth
* txtLength
* txtMassPerSquareMeter
* txtMaterial
* txtContractor
* txtStrengthCoefficient
* txtWeatherCondition
* txtSupervisor
* dateTimePickerCompletionDate
* listBoxRoadWorks

**Листинг программы (если есть)**

MainForm.cs

using System;

using System.Windows.Forms;

namespace zd6\_v2\_Menshikov

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeComponent();

listBoxRoadWorks.HorizontalScrollbar = true;

}

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

{

try

{

if (!double.TryParse(txtWidth.Text, out double width) ||

!double.TryParse(txtLength.Text, out double length) ||

!double.TryParse(txtMassPerSquareMeter.Text, out double massPerSquareMeter))

{

MessageBox.Show("Пожалуйста, введите корректные числовые значения для ширины, длины и массы на кв. метр.");

return;

}

string material = txtMaterial.Text;

string contractor = txtContractor.Text;

if (string.IsNullOrEmpty(txtStrengthCoefficient.Text))

{

RoadWork roadWork = new RoadWork(width, length, massPerSquareMeter, material, contractor);

RoadWorkManager.AddRoadWork(roadWork);

}

else

{

if (!int.TryParse(txtStrengthCoefficient.Text, out int strengthCoefficient))

{

MessageBox.Show("Пожалуйста, введите корректное значение для коэффициента прочности.");

return;

}

string weatherCondition = txtWeatherCondition.Text;

string supervisor = txtSupervisor.Text;

DateTime completionDate = dateTimePickerCompletionDate.Value;

AdvancedRoadWork advancedRoadWork = new AdvancedRoadWork(width, length, massPerSquareMeter, material, contractor, strengthCoefficient, weatherCondition, supervisor, completionDate);

RoadWorkManager.AddRoadWork(advancedRoadWork);

}

UpdateRoadWorksList();

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}");

}

}

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

{

UpdateRoadWorksList();

}

private void UpdateRoadWorksList()

{

listBoxRoadWorks.Items.Clear();

foreach (var roadWork in RoadWorkManager.RoadWorks)

{

listBoxRoadWorks.Items.Add(roadWork.ToString());

}

}

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

{

if (listBoxRoadWorks.SelectedItem != null)

{

string selectedItem = listBoxRoadWorks.SelectedItem.ToString();

string key = selectedItem.Split(':')[0].Trim();

RoadWorkManager.RemoveRoadWork(key);

UpdateRoadWorksList();

}

}

}

}

RoadWork.cs

using System;

namespace zd6\_v2\_Menshikov

{

public class RoadWork

{

public double Width { get; set; }

public double Length { get; set; }

public double MassPerSquareMeter { get; set; }

public string Material { get; set; }

public string Contractor { get; set; }

public RoadWork(double width, double length, double massPerSquareMeter, string material, string contractor)

{

Width = width;

Length = length;

MassPerSquareMeter = massPerSquareMeter;

Material = material ?? throw new ArgumentNullException(nameof(material));

Contractor = contractor ?? throw new ArgumentNullException(nameof(contractor));

}

public double CalculateQuality()

{

return (Width \* Length \* MassPerSquareMeter) / 1000;

}

public void DisplayInfo()

{

Console.WriteLine($"Ширина: {Width} м, Длина: {Length} м, Масса на кв. метр: {MassPerSquareMeter} кг, Материал: {Material}, Подрядчик: {Contractor}");

}

public override string ToString()

{

return $"Ширина: {Width} м, Длина: {Length} м, Масса на кв. метр: {MassPerSquareMeter} кг, Материал: {Material}, Подрядчик: {Contractor}";

}

}

}

RoadWorkManager.cs

using System;

using System.Collections.Generic;

using System.Linq;

namespace zd6\_v2\_Menshikov

{

public static class RoadWorkManager

{

public static List<RoadWork> RoadWorks { get; set; } = new List<RoadWork>();

public static Dictionary<string, RoadWork> RoadWorkDictionary { get; set; } = new Dictionary<string, RoadWork>();

public static void AddRoadWork(RoadWork roadWork)

{

RoadWorks.Add(roadWork);

}

public static void AddRoadWork(RoadWork roadWork, string key)

{

RoadWorkDictionary.Add(key, roadWork);

}

public static void AddRoadWork(List<RoadWork> roadWorks)

{

RoadWorks.AddRange(roadWorks);

}

public static void RemoveRoadWork(RoadWork roadWork)

{

RoadWorks.Remove(roadWork);

}

public static void RemoveRoadWork(string key)

{

if (RoadWorkDictionary.ContainsKey(key))

{

RoadWorkDictionary.Remove(key);

}

}

public static void RemoveRoadWork(List<RoadWork> roadWorks)

{

foreach (var roadWork in roadWorks)

{

RoadWorks.Remove(roadWork);

}

}

public static List<RoadWork> GetHighQualityRoadWorks(double threshold)

{

return RoadWorks.Where(r => r.CalculateQuality() > threshold).ToList();

}

}

}

AdvancedRoadWork.cs

using System;

namespace zd6\_v2\_Menshikov

{

public class AdvancedRoadWork : RoadWork

{

public int StrengthCoefficient { get; set; }

public string WeatherCondition { get; set; }

public string Supervisor { get; set; }

public DateTime CompletionDate { get; set; }

public AdvancedRoadWork(double width, double length, double massPerSquareMeter, string material, string contractor, int strengthCoefficient, string weatherCondition, string supervisor, DateTime completionDate)

: base(width, length, massPerSquareMeter, material, contractor)

{

StrengthCoefficient = strengthCoefficient;

WeatherCondition = weatherCondition ?? throw new ArgumentNullException(nameof(weatherCondition));

Supervisor = supervisor ?? throw new ArgumentNullException(nameof(supervisor));

CompletionDate = completionDate;

}

public new double CalculateQuality()

{

double baseQuality = base.CalculateQuality(); // Базовая формула качества

double qualityWithCoefficient;

if (StrengthCoefficient >= 5 && StrengthCoefficient <= 8)

{

qualityWithCoefficient = baseQuality \* 1.1; // Qp = Q \* 1.1

}

else if (StrengthCoefficient == 3 || StrengthCoefficient == 4 || StrengthCoefficient == 9 || StrengthCoefficient == 10)

{

qualityWithCoefficient = baseQuality \* 1.6; // Qp = Q \* 1.6

}

else

{

qualityWithCoefficient = baseQuality \* 1.9; // Qp = Q \* 1.9

}

return qualityWithCoefficient;

}

public override string ToString()

{

return base.ToString() + $", Коэффициент прочности: {StrengthCoefficient}, Погодные условия: {WeatherCondition}, Надзирающий: {Supervisor}, Дата завершения: {CompletionDate.ToShortDateString()}, Качество с учетом коэффициента: {CalculateQuality()}";

}

public void DisplayAdvancedInfo()

{

Console.WriteLine(this.ToString());

}

}

}

**Тестовые ситуации**

