БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Международный институт дистанционного обучения

Кафедра «Информационные системы и технологии»

**КОНТРОЛЬНАЯ РАБОТА**

по учебной дисциплине

«ОБЪЕКТНО-ОРИЕНТИРОВАННЫЕ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ И СТАНДАРТЫ ПРОЕКТИРОВАНИЯ»

Вариант № 3

РУКОВОДИТЕЛЬ: ст. преподаватель Русак Леонид Владимирович

ИСПОЛНИТЕЛЬ: студент 2119 учебной группы Мотолыга Д.И.

Минск 2021

Содержание

[**Задание № 1** 3](#_Toc86157478)

[**Исходный код** 4](#_Toc86157479)

[**Скриншоты работы приложения** 8](#_Toc86157480)

[**Заключение** 9](#_Toc86157481)

[**Задание № 2** 10](#_Toc86157482)

[**Исходный код** 11](#_Toc86157483)

[**Скриншоты работы приложения** 14](#_Toc86157484)

[**Заключение** 15](#_Toc86157485)

[**Список литературы** 16](#_Toc86157486)

# **Задание № 1**

Создать абстрактный класс Vehicle. На его основе реализовать классы Car (автомобиль), Bicycle (велосипед) и Lorry (грузовик). Классы должны иметь возможность задавать и получать параметры средств передвижения (цена, максимальная скорость, год выпуска и т.д.). Наряду с общими полями и методами, каждый класс должен содержать и специфичные для него поля.

Создать класс Garage, содержащий параметризованную коллекцию объектов этих классов в динамической памяти. Предусмотреть возможность вывода характеристик объектов списка. Написать демонстрационную программу, в которой будут использоваться все методы классов.

Дополнить класс методами сортировки по некоторому критерию, вывода в файл и считывания из файла.

## **Исходный код**

Абстрактный класс **Vehicle,** содержащий пять полей, конструктор и виртуальный метод getVehicleConfig.

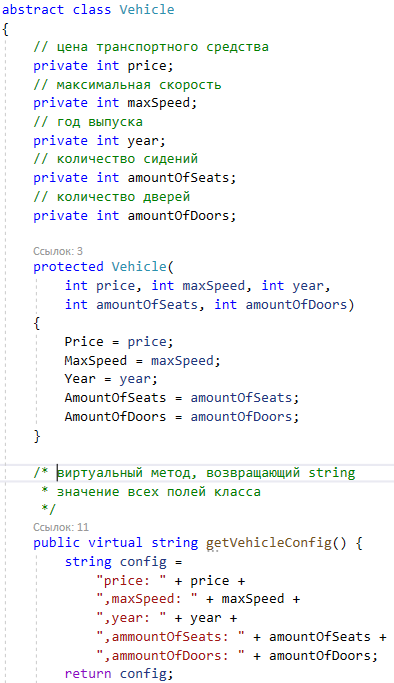


Рисунок 1 — класс Vehicle

Классы Car, Lorry, Bicycle, наследующие абстрактный класс Vehicle и переопределяющие виртуальный метод getVehicleConfig.

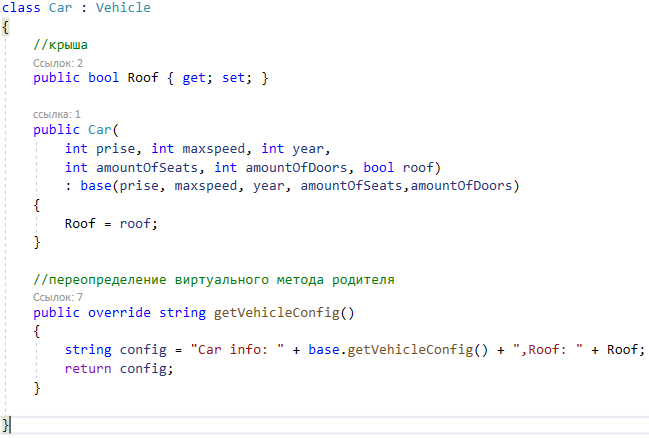


Рисунок 2 — класс Car

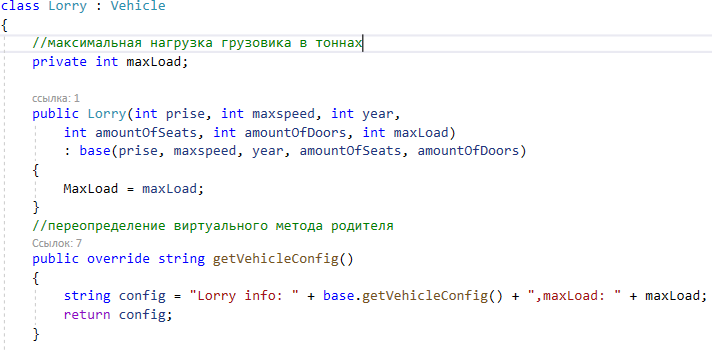


Рисунок 3 — класс Lorry

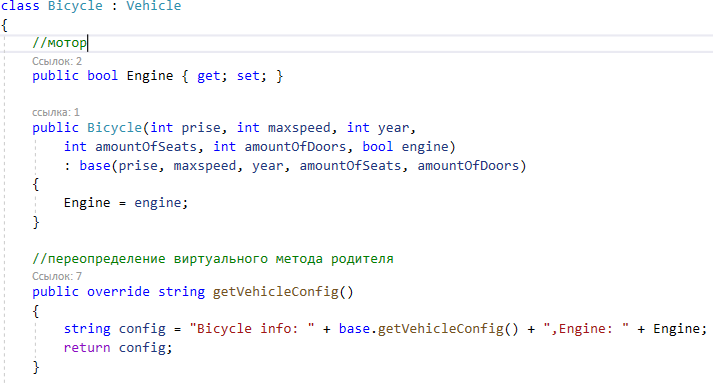


Рисунок 4 — класс Bicycle

А также класс Garage, в котором хранится параметризованная коллекция и методы по работе с ней: добавление, чтение в файл, чтение из файла и т.д.

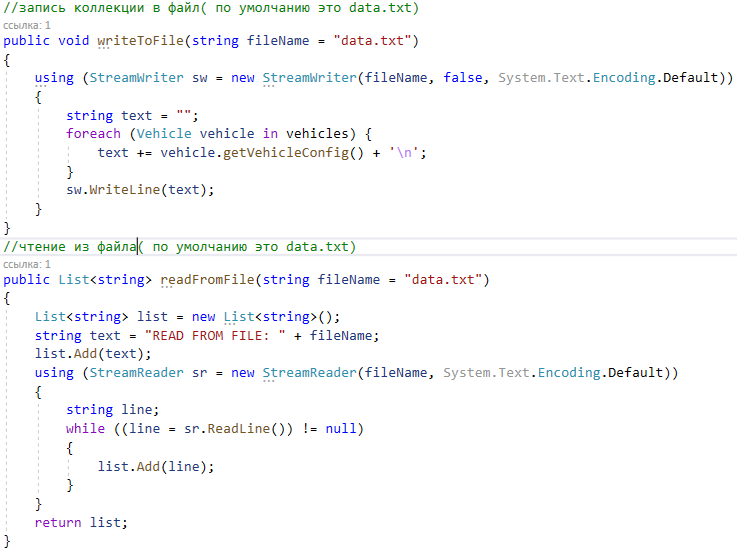
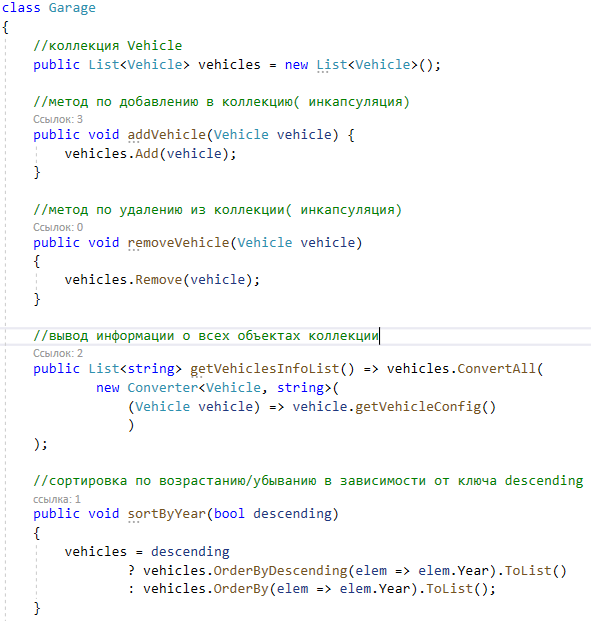


Рисунок 5 — класс Garage

## **Скриншоты работы приложения**

В приложении следующий функционал:

* Кнопки «add» - добавление в коллекцию объектов классов Car, Lorry, Bicycle;
* Кнопка «clearOutputGarage» - очистка текстового поля Garage;
* Кнопка «PrintGarageInMemory» - очистка коллекции в памяти;
* Кнопка «PrintGarageFromFile» - чтение из файла и вывод в текстовое поле Garage;
* Кнопка «WriteGarageToFile» - преобразование объекта Vehicle в строку и запись коллекции в файл;
* Кнопка «SortByYear» - в зависимости от чекбокса выполняет сортировку по убыванию/по возрастанию и выводит в текстовое поле Garage.

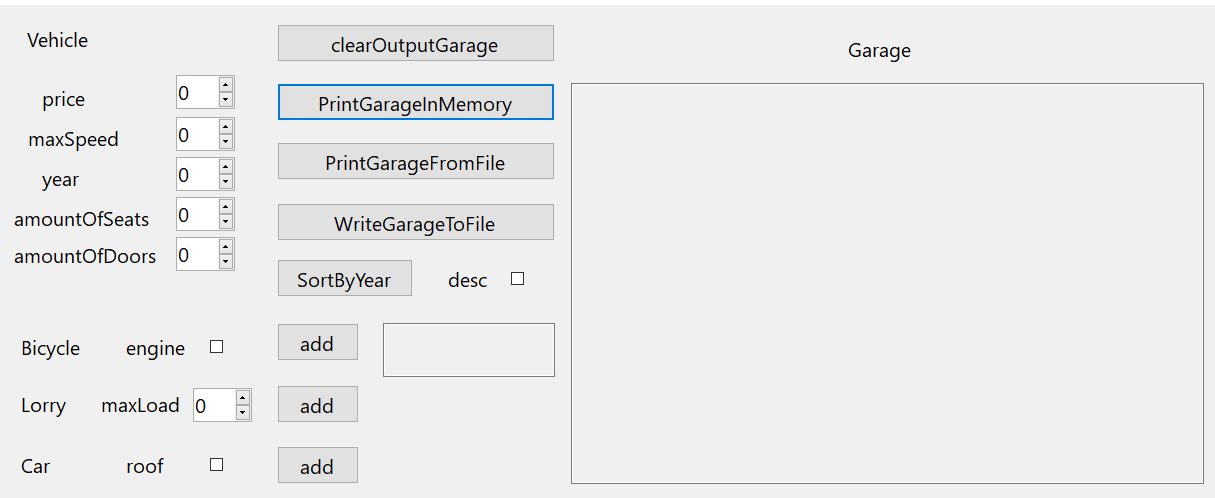


Рисунок 6 — Приложение задания 1

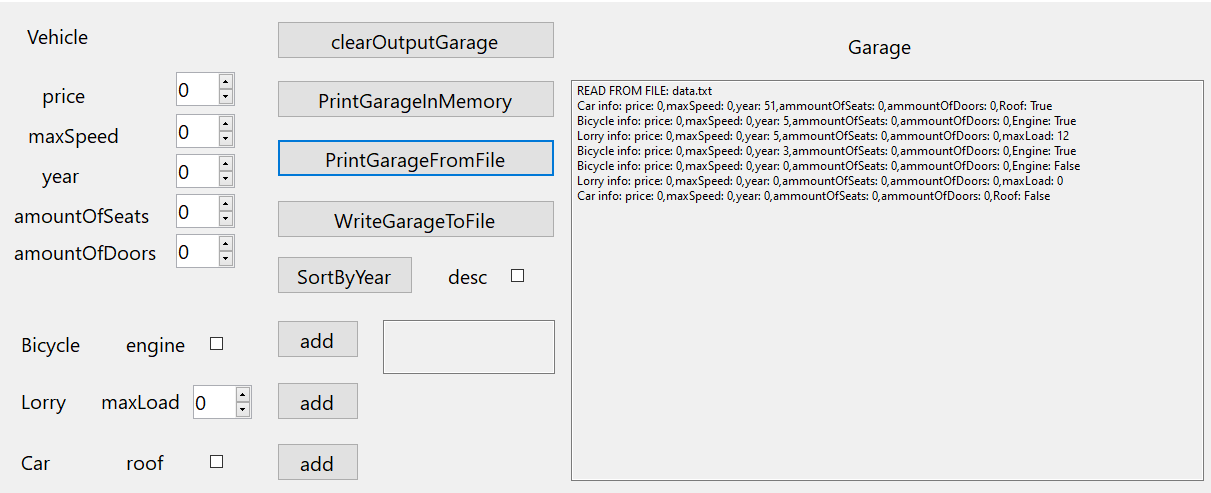


Рисунок 7 — Пример работы кнопки PrintGarageFromFile

## **Заключение**

В этом задании были закреплены знания в области ООП, а именно было продемонстрированно использование самых главных основ: инкапсуляции, наследования и полиморфизма.

Так же было разработано приложение в Windows Forms для удобства просмотра результатов, были обработаны исключительные ситуации.

# **Задание № 2**

Класс "Город". Поля – название, дата основания, площадь, к-во жителей, страна (перечисление). Создать метод, переводящий всю информацию в строку. Сохранить объект, созданный в main, в текстовый файл, используя предыдущий метод.

Разработать для своего класса метод, который выводит на экран всю информацию о классе и о полях класса вместе с их Hash-кодами.

## **Исходный код**

В этом задании был реализован класс City с шестью полями и перечисление Country – список стран.

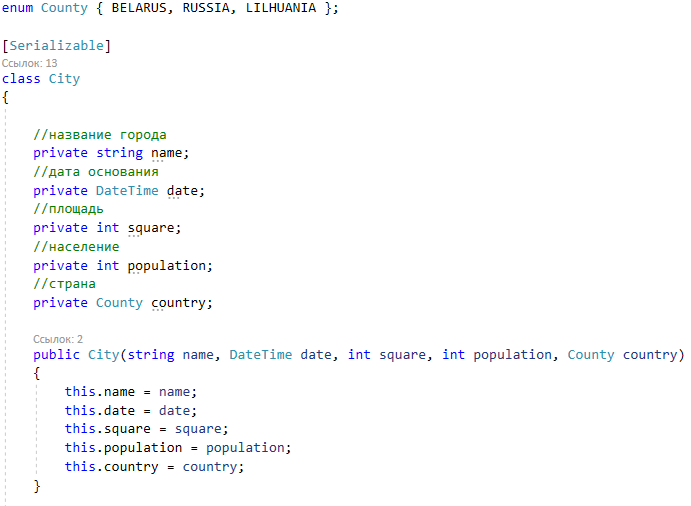


Рисунок 8 — Реализация класса City

Так же были реализованы методы записи в файл и чтения из файла путем парсинга.

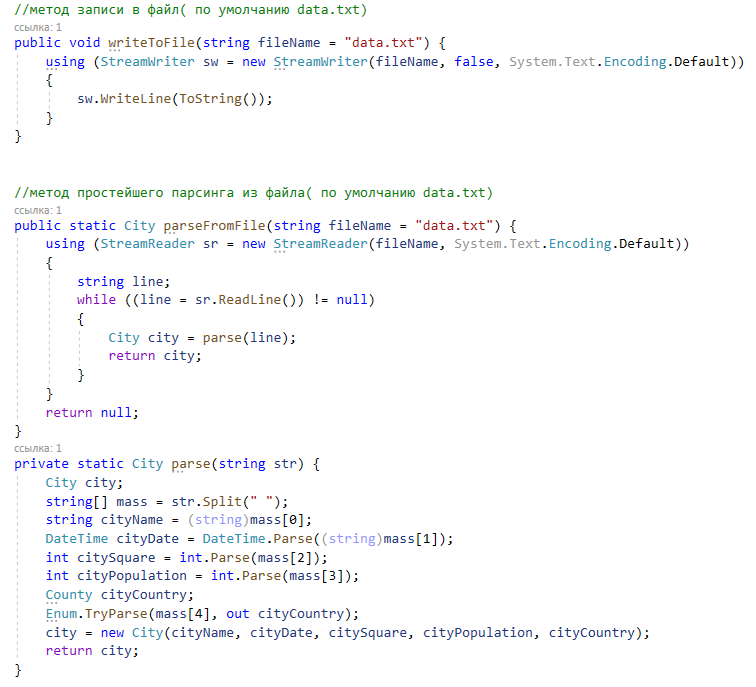


Рисунок 9 — Чтение и запись в файл

Так же была реализована сериализация и десириализация.



Рисунок 10 — Сериализация и десериализация

Так же по заданию необходимо было создать методы преобразования объекта в строку и в строку с Hash-кодами полей.



Рисунок 11 — Методы преобразования объекта

## **Скриншоты работы приложения**

В приложении следующий функционал:

* Кнопки «create» - создания объекта City;
* Кнопка «writeToFile» - запись в файл в виде строки;
* Кнопка «parseFromFile» - чтение и парсинг из файла в объект City;
* Кнопка «serialize» - сериализация в файл;
* Кнопка «deserialize» - десериализация из файла в объект City.

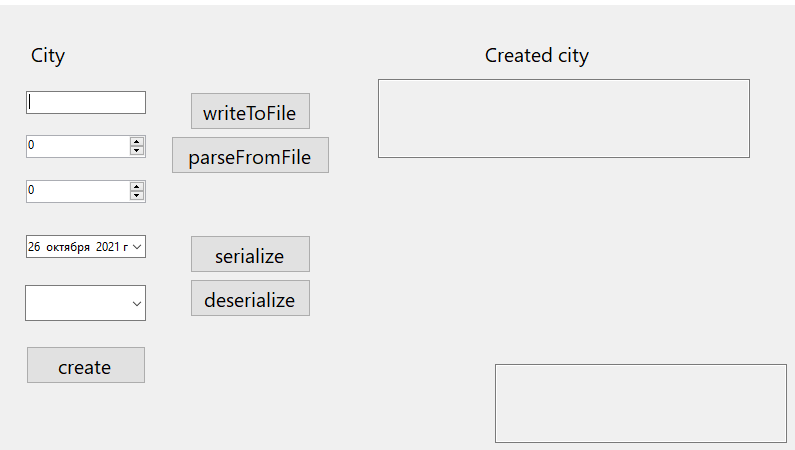


Рисунок 12 — Приложение задания 2

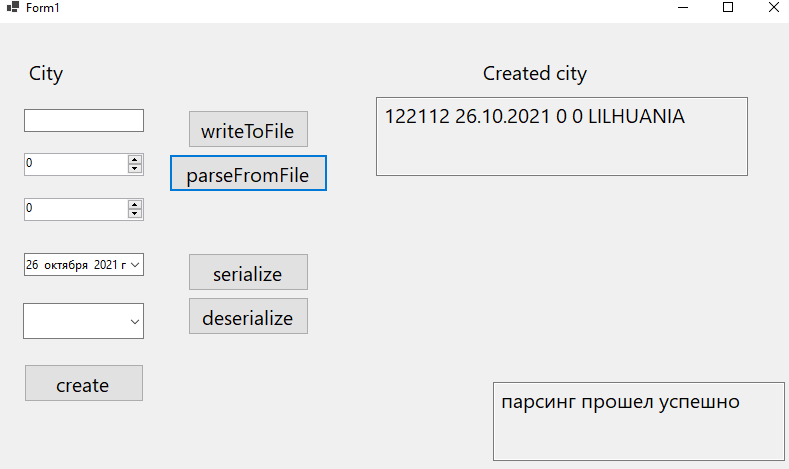


Рисунок 13 — Пример работы кнопки parseFromFile

## **Заключение**

В этом задании были закреплены знания в области записи и чтения из файла. Был разработан простейший парсер, а так же написан сериализатор и десериализатор.

Так же было разработано приложение в Windows Forms для удобства просмотра результатов, были обработаны исключительные ситуации.

# **Список литературы**

1. Шилдт, Герберт С# 4.0 Полное руководство. : Пер. с английского. М. – ООО «И.Д. Вильямс», 2011 – 1056 с. : ил.
2. Уотсон, К. Visual С# 2010: полный курс.: Пер. с англ. - М.: ООО "И.Д. Вильяме", 2011. - 960 с. : ил.
3. Троелсен, Эндрю. Язык программирования С# 2010 и платформа .NET 4.0, 5-е изд. : Пер. с англ. — М. : ООО "И.Д. Вильяме", 2011. — 1392 с. : ил.