Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №1

по дисциплине: «Шаблоны проектирования»

на тему «Порождающие паттерны»

Выполнил студент группы 19ВВП1:

Гусев Д. О.

Кубасов И.М.

Приняли:

Слепцов Н.В.

Пенза 2023

**Лабораторное задание:**

**Фабричный метод.**

Проект “Фабрика автомобилей”. В проекте должно быть реализовано создание нескольких марок автомобилей с различными характеристиками двигателей, кузова и комплектации.

**Листинг:**

**ICar**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab2.Cars

{

public interface ICar

{

string Brand { get; }

string Engine { get; set; }

string Body { get; set; }

string Equipment { get; set; }

}

}

**BMW**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab2.Cars

{

public class BMW : ICar

{

private string \_brand;

private string \_engine;

private string \_body;

private string \_equipment;

public BMW(string engine, string body, string equipment)

{

\_brand = "BMW";

\_engine = engine;

\_body = body;

\_equipment = equipment;

}

public string Brand { get { return \_brand; } }

public string Engine { get { return \_engine; } set { } }

public string Body { get { return \_body; } set { } }

public string Equipment { get { return \_equipment; } set { } }

}

}

**Ford**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab2.Cars

{

public class Ford : ICar

{

private string \_brand;

private string \_engine;

private string \_body;

private string \_equipment;

public Ford(string engine, string body, string equipment)

{

\_brand = "Ford";

\_engine = engine;

\_body = body;

\_equipment = equipment;

}

public string Brand { get { return \_brand; } }

public string Engine { get { return \_engine; } set { } }

public string Body { get { return \_body; } set { } }

public string Equipment { get { return \_equipment; } set { } }

}

}

**Nissan**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab2.Cars

{

public class Nissan : ICar

{

private string \_brand;

private string \_engine;

private string \_body;

private string \_equipment;

public Nissan(string engine, string body, string equipment)

{

\_brand = "Nissan";

\_engine = engine;

\_body = body;

\_equipment = equipment;

}

public string Brand { get { return \_brand; } }

public string Engine { get { return \_engine; } set { } }

public string Body { get { return \_body; } set { } }

public string Equipment { get { return \_equipment; } set { } }

}

}

**CarFabric**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using lab2.Cars;

namespace lab2.Factory

{

public abstract class CarFabric

{

public abstract ICar GetCar();

}

}

**BMWFactory**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using lab2.Cars;

namespace lab2.Factory

{

public class BMWFactory : CarFabric

{

private string \_engine;

private string \_body;

private string \_equipment;

public BMWFactory(string engine, string body, string equipment)

{

\_engine = engine;

\_body = body;

\_equipment = equipment;

}

public override ICar GetCar()

{

BMW car = new(\_engine, \_body, \_equipment)

{

Engine = \_engine,

Body = \_body,

Equipment = \_equipment,

};

return car;

}

}

}

**FordFactory**

using lab2.Cars;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab2.Factory

{

public class FordFactory : CarFabric

{

private string \_engine;

private string \_body;

private string \_equipment;

public FordFactory(string engine, string body, string equipment)

{

\_engine = engine;

\_body = body;

\_equipment = equipment;

}

public override ICar GetCar()

{

Ford car = new(\_engine, \_body, \_equipment)

{

Engine = \_engine, Body = \_body, Equipment = \_equipment,

};

return car;

}

}

}

**NissanFactory**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using lab2.Cars;

namespace lab2.Factory

{

public class NissanFactory : CarFabric

{

private string \_engine;

private string \_body;

private string \_equipment;

public NissanFactory(string engine, string body, string equipment)

{

\_engine = engine;

\_body = body;

\_equipment = equipment;

}

public override ICar GetCar()

{

Nissan car = new(\_engine, \_body, \_equipment)

{

Engine = \_engine,

Body = \_body,

Equipment = \_equipment,

};

return car;

}

}

}

**Program**

using lab2.Cars;

using lab2.Factory;

class Program

{

static void Main(string[] args)

{

List<ICar> cars = new List<ICar>();

FordFactory ford = new FordFactory("4000 лс", "прямой", "аптечка");

NissanFactory nissan = new NissanFactory("5000 лс", "косой", "навигатор");

BMWFactory bmw = new BMWFactory("6000 лс", "спорткар", "купленые права");

ICar car1 = ford.GetCar();

ICar car2 = nissan.GetCar();

ICar car3 = bmw.GetCar();

cars.Add(car1);

cars.Add(car2);

cars.Add(car3);

foreach (var car in cars)

{

Console.WriteLine("Марка: " + car.Brand);

Console.WriteLine("Двигатель: " + car.Engine);

Console.WriteLine("Кузов: " + car.Body);

Console.WriteLine("Комплектация: " + car.Equipment);

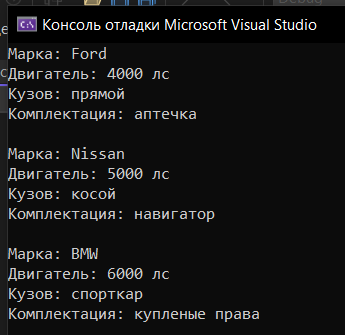
Console.WriteLine();

}

}

}

**Результаты работы:**



**Вывод:**

Научились применять на практике паттерн Фабричный метод.