**Лабораторная работа №12. Часть 2**

*«Классы. Наследование»*

**Рекомендации по программированию**

* Поля обязательно делать private.
* Интерфейс должен быть полным, т.е. предоставлять возможность выполнять любые разумные действия с классом. И одновременно минимально необходимым, т.е. без дублирования и пересечения возможностей методов.
* Каждый метод класса должен решать только одну задачу. Не стоит объединять два коротких независимых фрагмента кода в один метод.
* Если действия встречаются хотя бы дважды, стоит оформить их в отдельные функции.
* Все заданные методы, поля и экземпляры классов должны иметь осмысленные имена.

**Требования к лабораторной**

* Каждый разрабатываемый класс должен, как правило, содержать следующие элементы: поля с различными спецификаторами, конструкторы с параметрами и без параметров, методы, свойства.
* Методы и свойства должны обеспечивать непротиворечивый, полный, минимальный и удобный интерфейс класса.
* К каждому отчету, помимо формулировки задания, кода и скрина консоли, нужно прикрепить еще диаграмму классов

**Постановка задачи**

Реализовать иерархию классов.

1. Класс из лабораторной работы №11 Часть 2 сделать **абстрактным** (должно быть слово **abstract**), в котором поля и методы будут использоваться для последующих наследников.

Пример: *Класс Покемон с полями имя, пол, вес, рост и др.*

(Соответственно в классе Псайдак указанные поля для прошлой лабораторной, что будут едины для всех наследников класса Покемон, определять не нужно)

1. От абстрактного класса породить **еще несколько** (более двух) новых классов (помимо уже существующего). Добавить новые поля, определить для них методы абстрактного класса, добавить конструкторы, использующие конструктор абстрактного класса-родителя (через ключевое слово **base**), еще два-три новых поля и метода, один из которых **виртуальный**. Продемонстрировать работу каждого метода в консоли. Можно с использованием массивов (листов) персонажей.

Пример: *Класс Пикачу с дополнительными полями удар молнии, прыжок и др.*

*Класс Слоупок с дополнительными полями замедление, вызов дождя (это оказывается канон :с) и др.*

*Класс Псайдак (из прошлого задания с соответствующими полями).*

1. От каждого созданного производного класса построить **еще несколько** (минимум два наследника), расширяющие возможности этих классов. Определить конструкторы, использующие конструктор класса-родителя (через ключевое слово **base**), добавить новые поля и реализовать виртуальные методы класса-родителя (через ключевое слово **override**). Продемонстрировать работу каждого метода в консоли. Можно с использованием массивов (листов) персонажей.

Пример: *Наследники класса Пикачу: Пичу (с дополнительным полем умиление) и Райчу (с дополнительным полем передвижение на хвосте).*

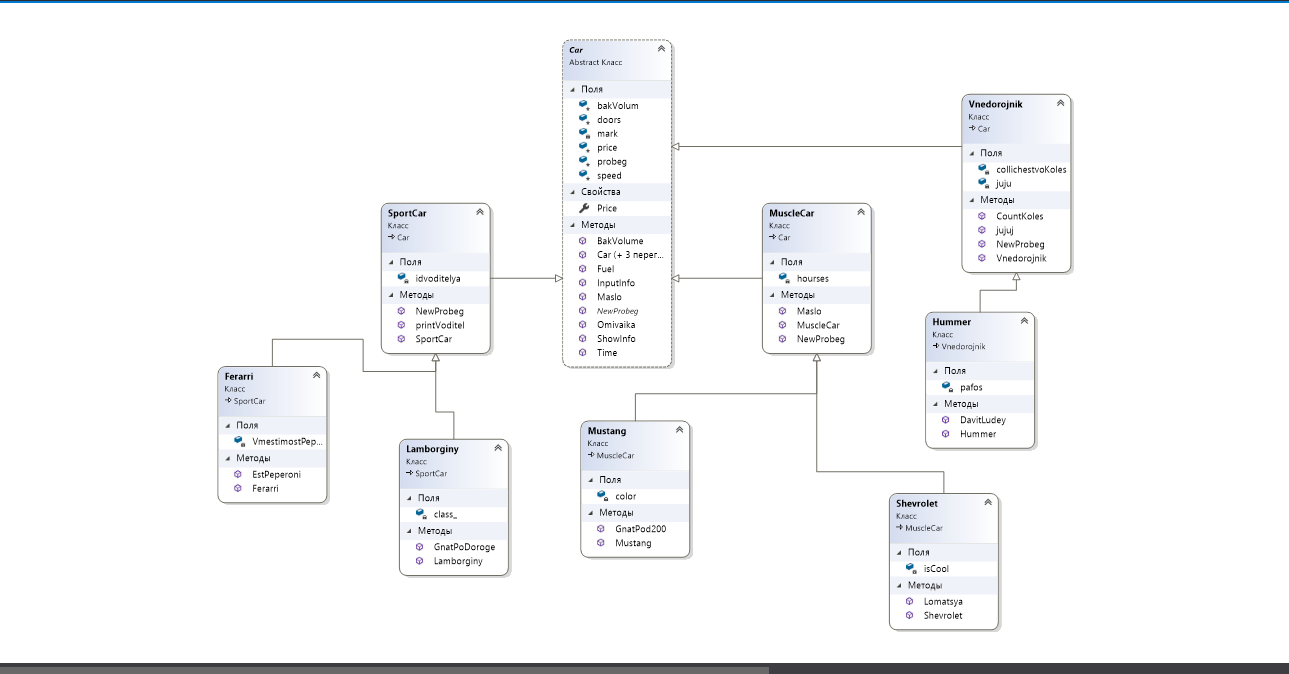
*Наследники класса Слоупок: Слоубро (с дополнительным полем… ну… командная медлительность?) и Слоукинг (с дополнительным полем медлительности на всех участников боя (включая и союзников :с)).*

*+ наследники класса Псайдак.*

Псы: Совет. Чаще всего виртуальным методом полезно делать метод вывода на экран каких-либо значений или результата для персонажей в иерархии. Но вы можете делать и какой-нибудь еще :)

Псы 2: Написание кода на русском все еще забавно)

Псы 3: Оценка зависит от количества реализованных классов. 10 классов – 10 баллов!



abstract class Car

{

string mark;

protected int doors;

protected double price;

protected double speed;

protected double probeg;

protected int bakVolum;

public double Price { get { return price; } }

public Car(double price, double speed, double probeg, string mark, int doors, int bakVolum)

{

this.mark = mark;

this.price = price;

this.probeg = probeg;

this.speed = speed;

this.doors = doors;

this.bakVolum = bakVolum;

}

public Car(double price, string mark, int doors)

{

this.mark = mark;

this.price = price;

this.probeg = 0;

this.speed = 0;

this.doors = doors;

}

public Car()

{

this.mark = "";

this.price = 0;

this.probeg = 0;

this.speed = 0;

this.doors = 0;

}

public Car(double price)

{

this.mark = "";

this.price = price;

this.probeg = 0;

this.speed = 0;

this.doors = 0;

}

public void InputInfo()

{

this.mark = Console.ReadLine();

this.price = Convert.ToDouble(Console.ReadLine());

this.probeg = Convert.ToDouble(Console.ReadLine());

this.speed = Convert.ToDouble(Console.ReadLine());

this.doors = Convert.ToInt32(Console.ReadLine());

this.bakVolum = Convert.ToInt32(Console.ReadLine());

}

public double Time(int range)

{

return range / speed \* 60;

}

public double Fuel(int range, int fuel)

{

return range / fuel;

}

public abstract double NewProbeg(int range);

virtual public int Maslo(int range)

{

return range / 3000;

}

public int BakVolume(int range)

{

if (bakVolum==0)

{

return 0;

}

return range / bakVolum;

}

public int Omivaika(int range, int omivayka)

{

return range / omivayka;

}

public void ShowInfo()

{

Console.WriteLine(price + " " + speed + " " + probeg + " " + mark + " " + doors + " " + bakVolum + " " + price);

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab12.part2

{

class Ferarri : SportCar

{

int VmestimostPeperoni;

public Ferarri(double price, double speed, double probeg, string mark, int doors, int bakVolum, int idvoditelya, int VmestimostPeperoni) : base(price, speed, probeg, mark, doors, bakVolum, idvoditelya)

{

this.VmestimostPeperoni = VmestimostPeperoni;

}

public void EstPeperoni()

{

Console.WriteLine("Vi sieli Peperoni");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab12.part2

{

class Hummer : Vnedorojnik

{

int pafos;

public Hummer(double price, double speed, double probeg, string mark, int doors, int bakVolum, int collichestvoKoles, string juju, int pafos) : base(price, speed, probeg, mark, doors, bakVolum, collichestvoKoles, juju)

{

this.pafos = pafos;

}

public void DavitLudey()

{

Console.WriteLine("Oh no, you are killer!!!!");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab12.part2

{

class Lamborginy : SportCar

{

char class\_;

public Lamborginy(double price, double speed, double probeg, string mark, int doors, int bakVolum, int idvoditelya, char class\_) : base(price, speed, probeg, mark, doors, bakVolum, idvoditelya)

{

this.class\_ = class\_;

}

public void GnatPoDoroge()

{

Console.WriteLine("Faster!!! Faster!!!!!");

}

}

}

using lab11;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab12.part2

{

class MuscleCar : Car

{

int hourses;

public MuscleCar(double price, double speed, double probeg, string mark, int doors, int bakVolum, int hourses) : base(price, speed, probeg, mark, doors, bakVolum)

{

this.hourses = hourses;

}

public override double NewProbeg(int range)

{

return range;

}

public int Maslo(int range)

{

return range / 2000;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab12.part2

{

class Mustang : MuscleCar

{

string color;

public Mustang(double price, double speed, double probeg, string mark, int doors, int bakVolum, int hourses, string color) : base(price, speed, probeg, mark, doors, bakVolum, hourses)

{

this.color = color;

}

public void GnatPod200()

{

Console.WriteLine("Your speed 200");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab12.part2

{

class Shevrolet : MuscleCar

{

bool isCool;

public Shevrolet(double price, double speed, double probeg, string mark, int doors, int bakVolum, int hourses, bool isCool) : base(price, speed, probeg, mark, doors, bakVolum, hourses)

{

this.isCool = isCool;

}

public void Lomatsya()

{

Console.WriteLine("Oh, this car broke again!");

}

}

}

using lab11;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab12.part2

{

class SportCar : Car

{

int idvoditelya;

public SportCar(double price, double speed, double probeg, string mark, int doors, int bakVolum, int idvoditelya) : base(price, speed, probeg, mark, doors, bakVolum)

{

this.idvoditelya = idvoditelya;

}

public override double NewProbeg(int range)

{

return range \* 2;

}

public void printVoditel()

{

Console.WriteLine(idvoditelya);

}

}

}

using lab11;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab12.part2

{

class Vnedorojnik : Car

{

int collichestvoKoles;

string juju;

public Vnedorojnik(double price, double speed, double probeg, string mark, int doors, int bakVolum, int collichestvoKoles, string juju) : base(price, speed, probeg, mark, doors, bakVolum)

{

this.collichestvoKoles = collichestvoKoles;

this.juju = juju;

}

public override double NewProbeg(int range)

{

return range;

}

public int CountKoles()

{

int c = 0;

for (int i = 0; i < collichestvoKoles; i++)

{

c++;

}

return c;

}

public string jujuj()

{

return juju + juju;

}

}

}

