**11 ПОЛИМОРФИЗМ**

Задание 1. Составить программу с одним родительским классом и двумя потомками. Потомки должны содержать виртуальные функции. Создать виртуальную функцию выдачи результатов расчета методов на экран монитора с указанием названий и полей, и их значений соответствующего объекта. Составить тестирующую программу с выдачей протокола на экран монитора. При этом создать объекты базового и производных типов, используя полиморфный контейнер - массив ссылок базового класса на объекты базового и производных классов (количество объектов&gt; =5).

Составить программу с абстрактным родительским классом и двумя объектами - потомками. Для этого модифицировать задание 2. Составить тестирующую программу с выдачей протокола на экран монитора. В ней нужно реализовать циклический вывод параметров объектов, используя полиморфный контейнер - массив объектов базового класса (количество объектов&gt;=5).

Листинг программы:

using System;

namespace task1

{

class Car

{

public string name;

public Car()

{

this.name = null;

}

public virtual void Input()

{

Console.Write("Введиете название машины-");

name = Console.ReadLine();

}

public virtual void GetInfo()

{

Console.WriteLine($"название: {name}");

}

}

class Cargo:Car

{

public double liftingCapacityInT;

public double consumption;

public Cargo()

{

this.liftingCapacityInT = 0.0;

this.consumption = 0.0;

Input();

}

public override void Input()

{

base.Input();

Console.Write("Введите грузоподъемность в тоннах: ");

liftingCapacityInT = Convert.ToDouble(Console.ReadLine());

}

public virtual void Consumption()

{

consumption = Math.Sqrt(liftingCapacityInT) \* 100;

}

public override void GetInfo()

{

Consumption();

Console.WriteLine($"Название: {name}\nгрузоподъемность в тоннах: {liftingCapacityInT}\nрасход горючего на 100км: {consumption}");

}

}

class Passenger : Car

{

public double engineCapacity;

public double consumption;

public Passenger()

{

this.engineCapacity = 0.0;

this.consumption = 0.0;

Input();

}

public override void Input()

{

base.Input();

Console.Write("Введите объем двигателя в см кубических: ");

engineCapacity = Convert.ToDouble(Console.ReadLine());

}

public virtual void Consumption()

{

consumption = engineCapacity \* 2.5;

}

public override void GetInfo()

{

Consumption();

Console.WriteLine($"Название: {name}\nобъем двигателя в см кубических: {engineCapacity}\nрасход горючего на 100км: {consumption}");

}

}

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Выберите создаваемый объект:");

Console.WriteLine("1.Car\n2.Cargon\n3.Passenger");

int change = Convert.ToInt32(Console.ReadLine());

int i = 0;

double sum = 0;

Console.Write("Введите количество объектов: ");

int mascount = Convert.ToInt32(Console.ReadLine());

Car[] cars = new Car[mascount];

while (i< mascount)

{

switch (change)

{

case 1: cars[i] = new Car(); i++; break;

case 2: Cargo cargo = new Cargo();

cargo.Consumption();

sum += cargo.consumption;

cars[i] = cargo as Car;++i;break;

case 3: Passenger passenger = new Passenger();

passenger.Consumption();

sum += passenger.consumption;

cars[i] = passenger as Car;++i;break;

default: Console.WriteLine("Нет такого пункта!");break;

}

}

for (int t = 0; t < cars.Length; t++)

{

if (cars[t] != null)

{

cars[t].GetInfo();

}

}

Console.WriteLine("суммарный расход горючего на 100 км: "+ sum);

}

}

}

Таблица 11.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 2, 2, Porsche, 1, ford mustang, 1 | расход горючего на 100км: 100  суммарный расход горючего на 100 км: 200 |

Анализ результатов:

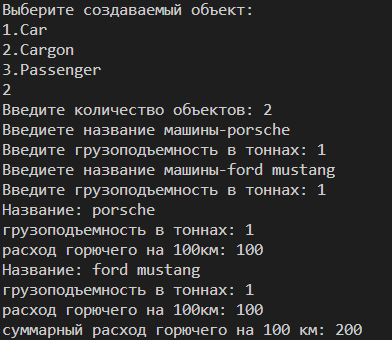


Рисунок 11.1 – Результат работы программы

Источник – собственная разработка