**10 Механизм наследования. Отношения между классами**

Задание 1. Составить программу с одним родительским классом и потомком. Все поля должны быть закрытыми. Базовый класс должен содержать конструкторы с параметрами, методы доступа к закрытым полям, вывод полей и указанный метод. Производный класс содержит дополнения и изменения, организовать вывод новых полей потомка, при этом имена методов совпадают с именами методов базового класса. Составить тестирующую программу с выдачей результатов. Создать объекты базового и производного типов. В программе должна выполняться проверка всех разработанных элементов класса. Базовый класс: Автобус (поля: количество пассажиров, стоимость билета) Метод: Общая стоимость всех мест Потомок: Скорый (поле – скорость, марка автобуса) Изменения в потомках: Найти общую стоимость всех мест с учетом увеличения цены билета на 0.05 скорости.

Листинг программы:

namespace MyNamespace

{

class Autobus

{

private int passengers;

private decimal ticketPrice;

public Autobus(int passengers, decimal ticketPrice)

{

this.passengers = passengers;

this.ticketPrice = ticketPrice;

}

public int Passengers

{

get { return passengers; }

set { passengers = value; }

}

public decimal TicketPrice

{

get { return ticketPrice; }

set { ticketPrice = value; }

}

public virtual decimal TotalCost()

{

return passengers \* ticketPrice;

}

}

class Skory : Autobus

{

private double speed;

private string brand;

public Skory(int passengers, decimal ticketPrice, double speed, string brand)

: base(passengers, ticketPrice)

{

this.speed = speed;

this.brand = brand;

}

public double Speed

{

get { return speed; }

set { speed = value; }

}

public string Brand

{

get { return brand; }

set { brand = value; }

}

public override decimal TotalCost()

{

decimal increasedTicketPrice = TicketPrice \* (decimal)(1 + 0.05 \* speed);

return Passengers \* increasedTicketPrice;

}

}

class Program

{

static void Main(string[] args)

{

Autobus bus = new Autobus(50, 10.0m);

Console.WriteLine($"Пассажиры автобуса: {bus.Passengers}\nЦена билета: {bus.TicketPrice}\nОбщая стоимость: {bus.TotalCost()}");

Console.WriteLine();

Skory expressBus = new Skory(50, 10.0m, 100.0, "Мерседес");

Console.WriteLine($"Пассажиры экспресс-автобуса: {expressBus.Passengers}\nСтоимость билета: {expressBus.TicketPrice}\nСкорость: {expressBus.Speed}\nМарка: {expressBus.Brand}\nОбщая стоимость: {expressBus.TotalCost()}");

}

}

}

Таблица 10.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 50, 10.0m  50, 10.0m, 100.0, "Мерседес" | 500  3000 |

Анализ результатов:

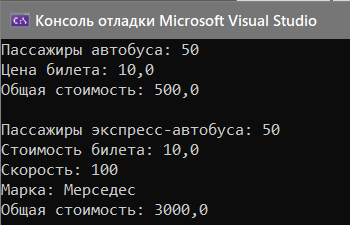


Рисунок 10.1 – Результат работы программы

Задание 2. В рамках консольного приложения разработать класс В- наследник класса А (класс А с полями а и b и свойством с. Свойство – значение выражения над полями а и b (выражение и типы полей). Поля инициализировать при объявлении класса. Конструктор оставить по умолчанию. Проследить, чтобы поля а и b напрямую в других классах были недоступны.) с полем d и свойством с2. Свойство с2 – результат вычисления выражения над полями a, b, d. В теле свойства использовать управляющий оператор Switch. Для класса В определить 2 конструктора: один – наследуется от конструктора класса А, второй –собственный. В теле программы создать объекты классов А и В, продемонстрировав работу всех конструкторов. Вывести значения свойства на экран.

Листинг программы:

namespace MyNamespace

{

class A

{

protected float a = 2.4F;

protected float b = 4.3F;

public float c

{

get

{

var per = a \* 30 / b;

per %= 10;

return per;

}

}

public A()

{

}

public A(float a1, float b1)

{

a = a1;

b = b1;

}

}

class B : A

{

public float d;

public float c2

{

get

{

switch ((int)d)

{

case 0:

return (a + b) \* 2;

default:

var per = a \* b;

per %= 10;

return per;

}

}

}

public B(float a1, float b1) : base(a1, b1)

{

a = a1;

b = b1;

d = 7.1F;

}

public B(float a1, float b1, float d1) : this(a1, b1)

{

a = a1;

b = b1;

d = d1;

}

}

class Program

{

static void Main(string[] args)

{

var classA = new A();

Console.WriteLine(classA.c);

var classA2 = new A(3.4F, 7.8F);

Console.WriteLine(classA2.c);

var classB = new B(3.4F, 7.8F);

Console.WriteLine(classB.c);

Console.WriteLine(classB.c2);

var classB2 = new B(3.4F, 7.8F, 0F);

Console.WriteLine(classB2.c);

Console.WriteLine(classB2.c2);

}

}

}

Таблица 10.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 3.4F, 7.8F  3.4F, 7.8F, 0F | 6,7441845  3,0769224  3,0769224  6,5200005  3,0769224  22,400002 |

Анализ результатов:

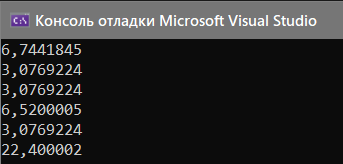


Рисунок 10.2 – Результат работы программы