**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: Студент (поле: имя, средний балл s). Метод: Стипендия 300000+10000([s]-5) Потомок: Магистр (поле – специальность). Изменения в потомках: Увеличить стипендию на m руб.

Листинг программы:

Magistor student = new Magistor("Talerchik", 9.0, "TP");

Console.WriteLine(student.AddStipend());

Console.WriteLine(student.AddStipend(5));

class Student

{

private string \_name;

private double \_s;

public string Name

{

get { return \_name; }

set { \_name = value; }

}

public double s {

get { return \_s; }

set { \_s = value; }

}

public double AddStipend()

{

return 3000000 + 10000 \* (\_s - 5); }

public Student(string nameData, double sData) {

\_name = nameData;

\_s = sData;

}

}

class Magistor : Student

{

private string \_speciality;

public string Speciality

{

get { return \_speciality; }

set { \_speciality = value; }

}

public Magistor(string nameData, double sData, string speciality)

: base(nameData, sData)

{

Speciality = speciality;

}

public double AddStipend(int m)

{

return (3000000 + 10000 \* (s - 5)) + m;

}

}

Таблица 10.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| \_name, \_s, \_specialty | 3040000, 3040005 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 10.1.



Рисунок 10.1 – Результат работы программы

Источник: собственная разработка

Задание №2. В рамках консольного приложения разработать класс B – наследник класса A (класс A с полями a и b и свойством c). Свойство – значение выражения над полями a и b. Поля инициализировать при объявлении класса. Конструктор оставить по умолчанию.

Проследить, чтобы поля a и b напрямую в других классах были недоступны. С полем d и свойством c2. Свойство c2 – результат вычисления выражения над полями a, b, d. В теле свойства использовать управляющий оператор While. Для класса B определить 2 конструктора: один – наследуется от конструктора класса A, второй – собственный. В теле программы создать объекты классов A и B, продемонстрировав работу всех конструкторов. Вывести значения свойства на экран.

Листинг программы:

class A

{

protected int a;

protected int b;

public A(int a, int b)

{

this.a = a;

this.b = b;

}

public A()

{

this.a = default(int);

this.b = default(int);

}

}

class B : A

{

private int d;

public B(int a, int b, int d)

: base(a, b)

{

this.d = d;

}

public B(int d)

{

this.d = d;

}

public int c2

{

get

{

var sum = 0;

while (a < 10)

{

sum += b + d;

a++;

}

return sum;

}

}

}

internal class Program

{

static void Main(string[] args)

{

B b = new B(0, 10, 2);

Console.WriteLine(b.c2);

Console.ReadLine();

}

}

Таблица 10.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 0, 10, 2 | 120 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 10.2.



Рисунок 10.2 – Результат работы программы

Источник: собственная разработка