**Механизм наследования. Отношения между классами**

**Задание №1.** 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:

static void Main(string[] args)

{

Constant constant = new();

var studentName = Faker.NameFaker.FirstName();

var averageScore = Faker.NumberFaker.Number(constant.GetMinConst(), constant.GetMaxConst()) \* 0.1;

Student student = new Student(studentName, averageScore);

var specialization = Faker.NameFaker.LastName();

Magistr magistr = new Magistr(specialization, Faker.NumberFaker.Number(constant.GetMinConst(), constant.GetMaxConst()), studentName, averageScore);

student.DetermScholarship();

WriteLine(student.GetAllInfo());

magistr.DetermScholarship();

WriteLine(magistr.GetAllInfo());

ReadKey();

}

Анализ результатов:

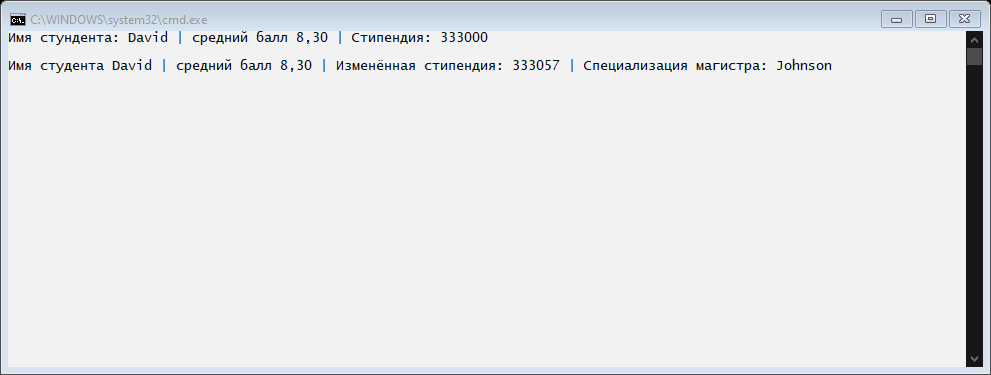


Рисунок 1.1 – Результат работы программы

**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. Поля инициализировать при объявлении класса. Конструктор оставить по умолчанию. Проследить, чтобы поля а и b напрямую в других классах были недоступны.). Класс B с полем d и свойством с2. Свойство с2 – результат вычисления выражения над полями a, b, d. В теле свойства использовать управляющий оператор. 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:

static void Main(string[] args)

{

Constant constant = new();

ClassA classA = new ClassA(Faker.NumberFaker.Number(constant.GetMinConst(), constant.GetMaxConst()),

Faker.NumberFaker.Number(constant.GetMinConst(), constant.GetMaxConst()));

ClassB classB = new ClassB(Faker.NumberFaker.Number(constant.GetMinConst(), constant.GetMaxConst()),

Faker.NumberFaker.Number(constant.GetMinConst(), constant.GetMaxConst()),

Faker.NumberFaker.Number(constant.GetMinConst(), constant.GetMaxConst()));

WriteLine($"CLASS\_A: a = {classA.GetA()} b = {classA.GetB()} | c = {classA.C()}");

WriteLine($"CLASS\_B: a = {classB.GetA()} b = {classB.GetB()} | d = {classB.GetD()} | c = {classB.C2()}");

ReadKey();

}

Анализ результатов:

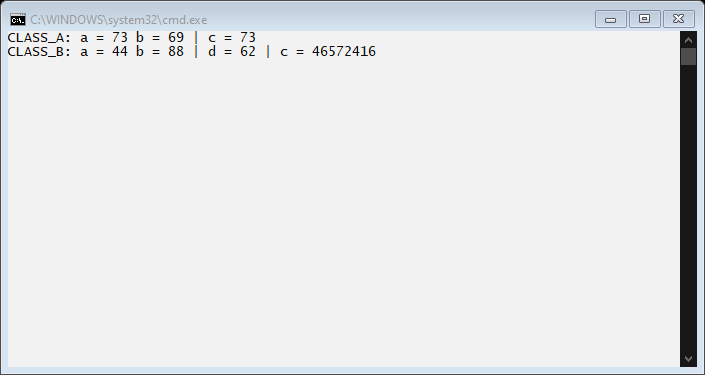


Рисунок 1.2 – Результат работы программы