Университет ИТМО

**Лабораторная работа #7**

**Создание иерархии классов**

Выполнил: Михайлов

Иван Юрьевич

Группа № К3121

Проверила: Филимонова И. А.

Санкт-Петербург

2020

**Цель работы:**

Рассмотреть парадигмы объектно-ориентированного программирования: полиморфизм, наследование.

**Ход работы:**

**Упражнение 1.** Реализация наследования классов.

Скопировано решение MyClass, из лабораторной работы №6. Добавлен класс Item. В него добавлены 2 поля со спецификатором доступа protector. Добавлены методы, реализующие работу с единицей хранения: IsAvailable, GetInvNumber, Take, Return, Print.

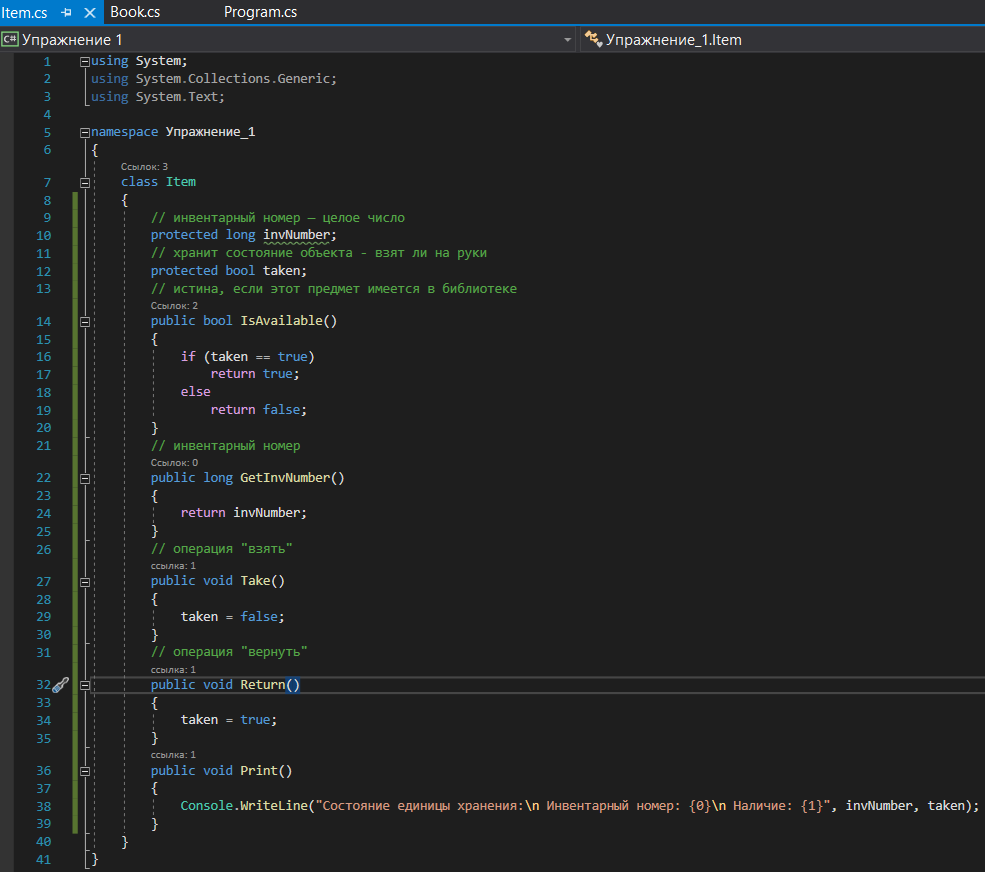


Рисунок 1 – Класс Item

Класс Book переведён в производный класс от класса Item и наследует все переменные и методы последнего. Чтобы не было конфликта между методами Print в базовом и производном классах, в производном классе явно указан факт скрытия метода базового класса. Также в класс Book добавлены методы выдачи и возврата книг.

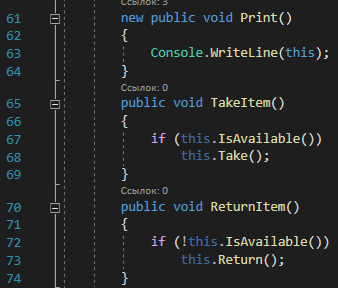


Рисунок 2 – Изменения в классе Book

В методе Main с помощью конструктора по умолчанию базового класса создан объект – единица хранения и на экране отображена её значение по умолчанию.

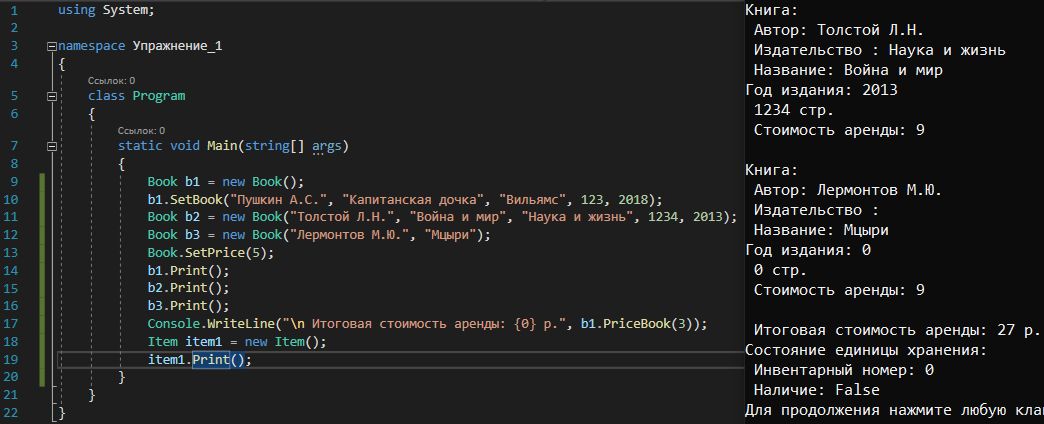


Рисунок 3 – Основная логика и результат работы программы

**Упражнение 2.** Использование конструкторов.

В базовый класс Item добавлены два конструктора: первый, с параметрами для инициализации объекта хранения при его создании; второй, по умолчанию для инициализации начальными значениями, учитывая, что в начале работы предполагается наличие элемента в библиотеке.

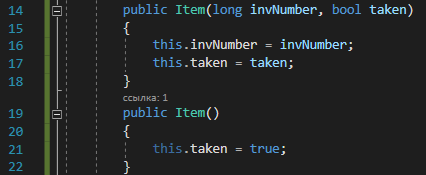


Рисунок 4 – Изменения в классе Item

В производный класс Book добавлен конструктор, который ссылается на конструктор базового класса с помощью ключевого слова base, принимающего параметры любого существующего конструктора базового класса.

Для вывода информации о тех полях, которые наследуются производным классом в методе производного класса вызван метод базового класса. Для этого применяется также ключевое слово base.

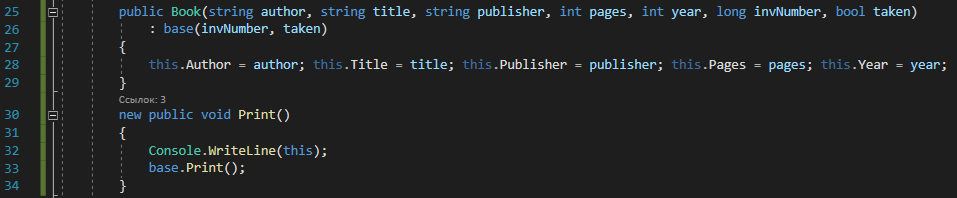


Рисунок 5 – Реализация наследования

При создании объекта b2 добавлены параметры базового класса. Также, перед выводом данных о ней, вызван метод выдачи книги читателю.

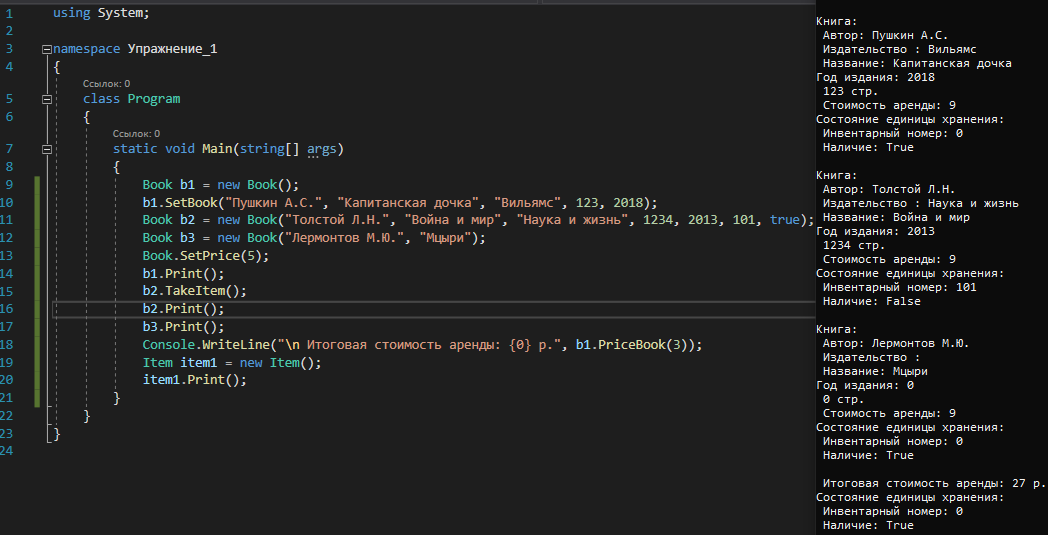


Рисунок 6 – Демонстрация функционирования наследования

Добавлен новый класс Magazine, также унаследованный от класса Item. В него добавлены 4 автоматических свойства для описания. Добавлены два конструктора: со всеми 6 параметрами и по умолчанию. Переопределён метод ToString с помощью ключевого слова override. Добавлен метод, отображающий информацию об объекте – журнале. Методы TakeItem и ReturnItem перенесены из класса Book в базовый класс Item, чтобы к ним имели доступ оба унаследованных класса. Методу Take указан модификатор доступа private, так как теперь он применяется только внутри своего класса.

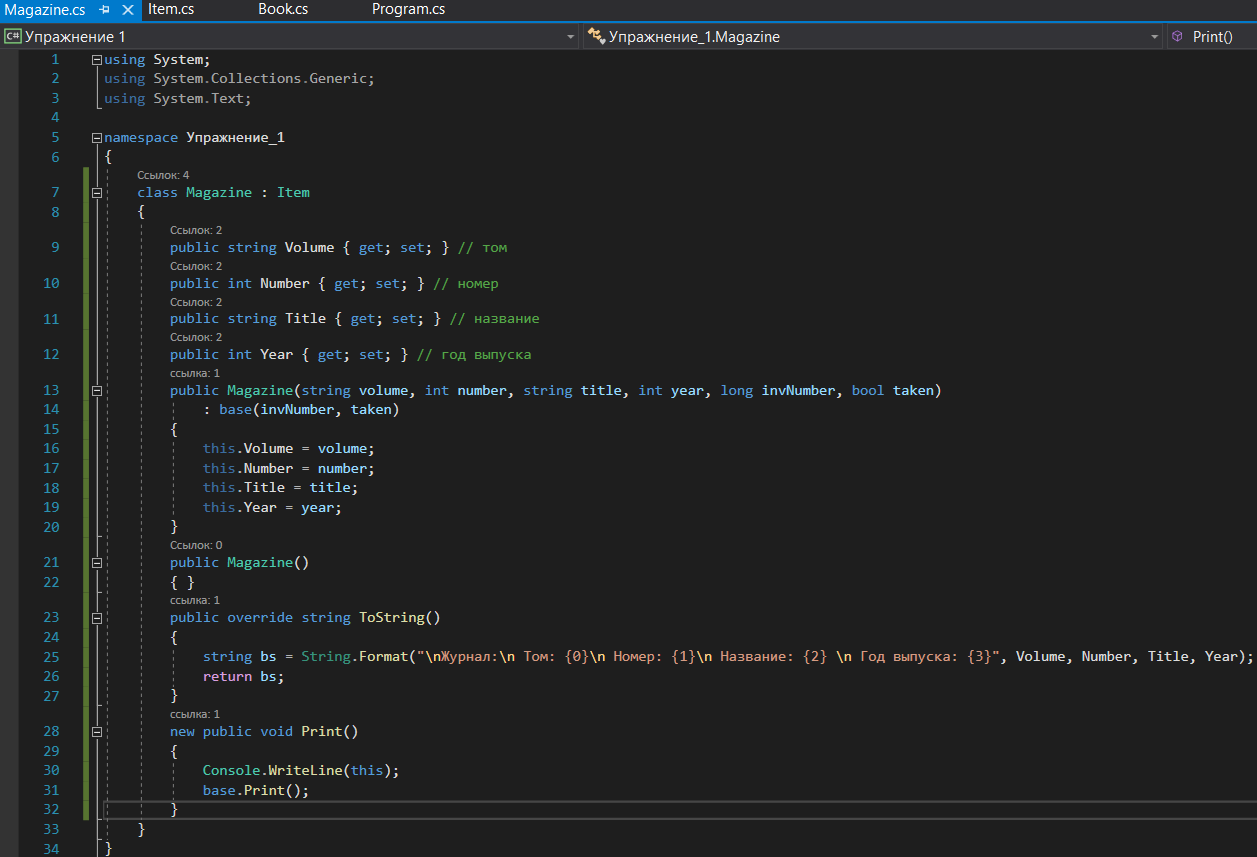


Рисунок 7 – Новый класс Magazine

В методе Main класса Program создан объект класса Magazine – журнал с передачей параметров в конструктор. Вызваны методы выдачи журнала и отображения информации о нём.

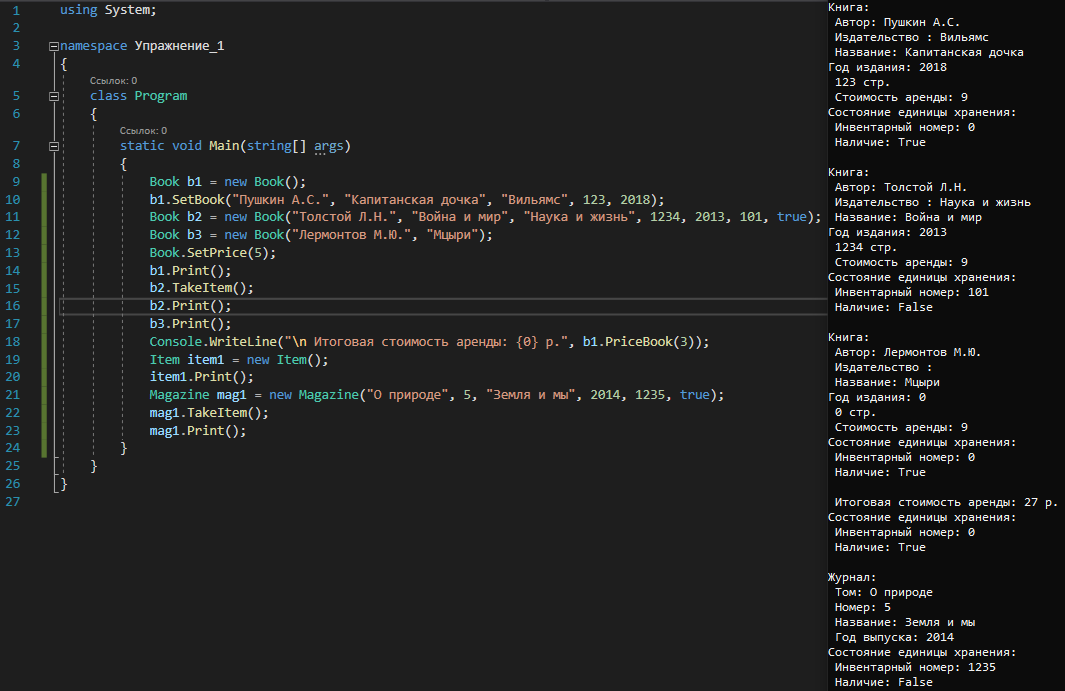


Рисунок 8 – Создание объекта Magazine

**Упражнение 3.** Переопределение методов.

В базовом классе методы Return и Print объявлены виртуальными, так как их реализация для объектов произвольных классов различна. В классах наследниках эти методы переопределены с помощью ключевого слова override. Помимо этого, в класс Book добавлено новое свойство булевого типа, причём доступ к его записи извне класса получить невозможно (private set). Добавлен метод ReturnSrok устанавливающий, что книга сдана в срок.

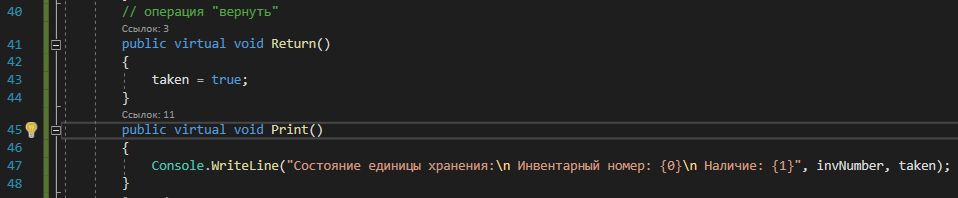


Рисунок 9 – Объявление методов виртуалными

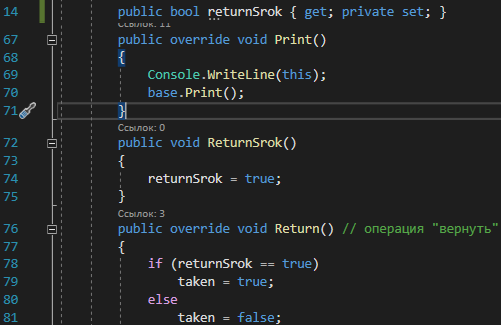


Рисунок 10 – Изменения в классе Book

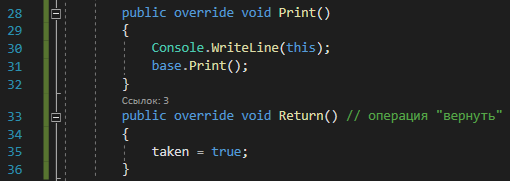


Рисунок 11 – Реализация полиморфизма

В методе Main класса Program объявлена ссылка на объект базового класса. Она ссылается на объект b2 класса Book. От её имени вызваны оба виртуальных метода. То же самое выполнено для объекта mag1 класса Magazine.

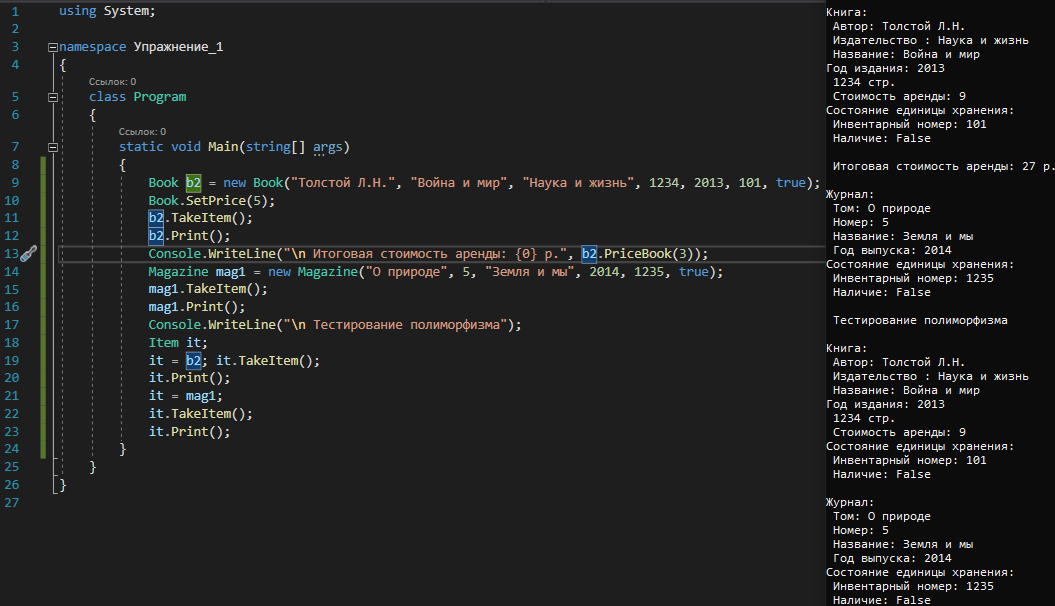


Рисунок 12 – Демонстрация функционирования полиморфизма

**Упражнение 4.** Применение абстрактного класса и абстрактных методов.

Класс Item реализован как абстрактный, с помощью ключевого слова abstract. В нём же реализован метод Return как абстрактный, из которого удалено всё тело. В методе Main класса Program после создания книги b2 вызван метод ReturnSrok() (книга возвращена в срок) и абстрактный метод Return() к объектам обоих классов.



Рисунок 13 – Реализация абстрактного класса

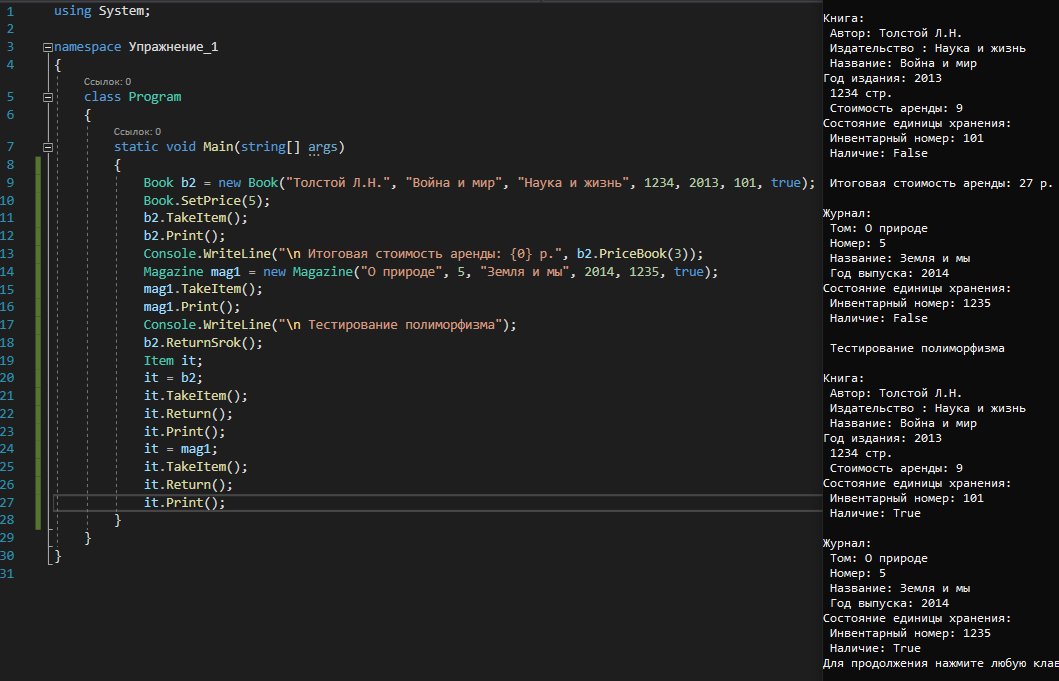


Рисунок 14 – Функционирование абстрактного метода

**Упражнение 5.** Реализация отношения ассоциации между классами.

Добавлен класс Publisher. В него добавлены три поля: имя строкового типа, имейл-адресс строкового типа, закрытый номер лицензии структурного типа LicenseNumber. Определена структура LicenseNumber с двумя полями: номером лицензии целочисленного типа, дата выдачи временного типа данных; и переопределённым методом ToString(). Также в классе определён конструктор и переопределён по-своему метод ToString().

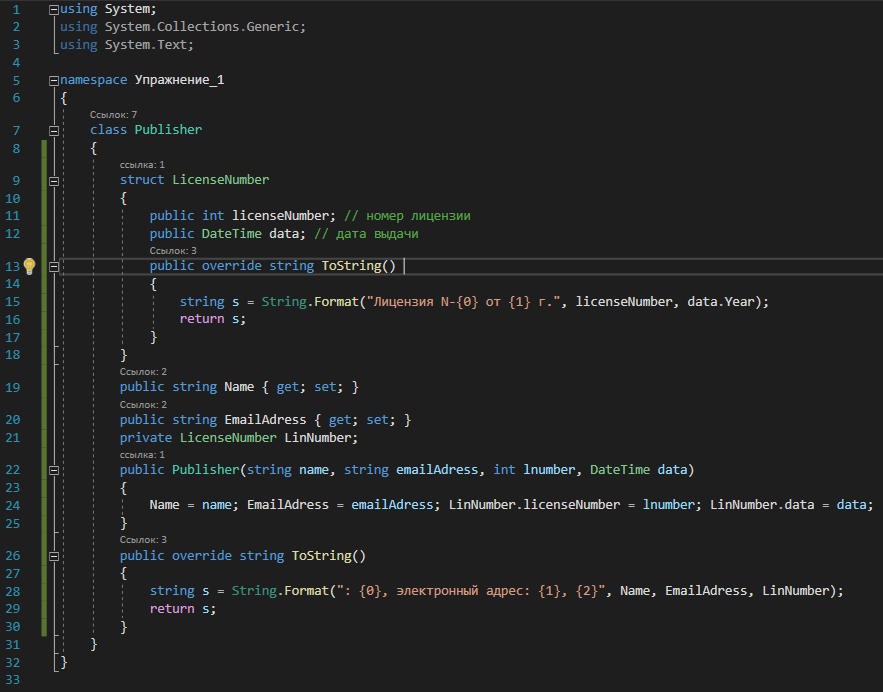


Рисунок 15 – Новый класс Publisher

В классе Book заменено объявление свойства издательства на ссылку типа Publisher. В соответствии с этим изменены конструкторы класса и методы ToString() и SetBook().

В методе Main класса Program до создания объекта b2 создан объект издательства. При создании объекта книги одним из полей передаётся объект издательства.

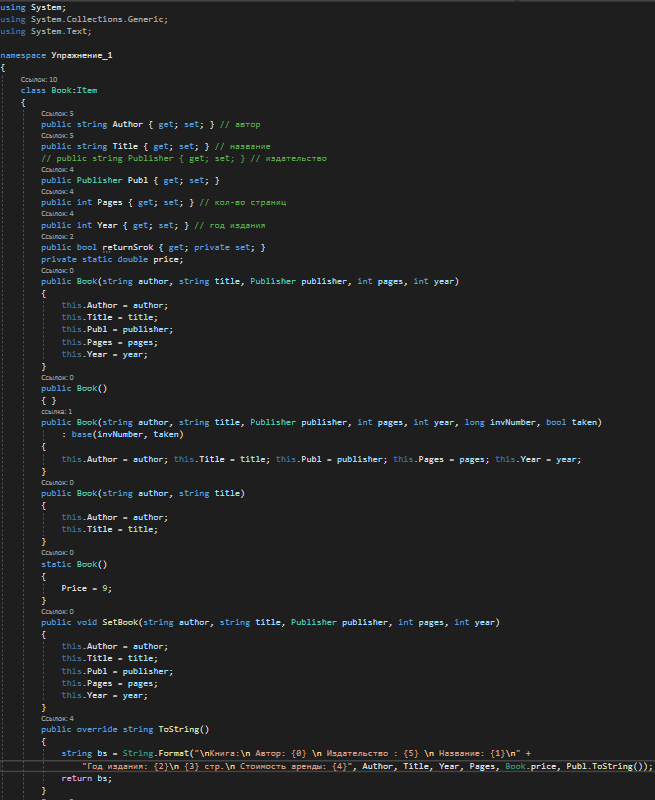


Рисунок 16 – Изменения в классе Book

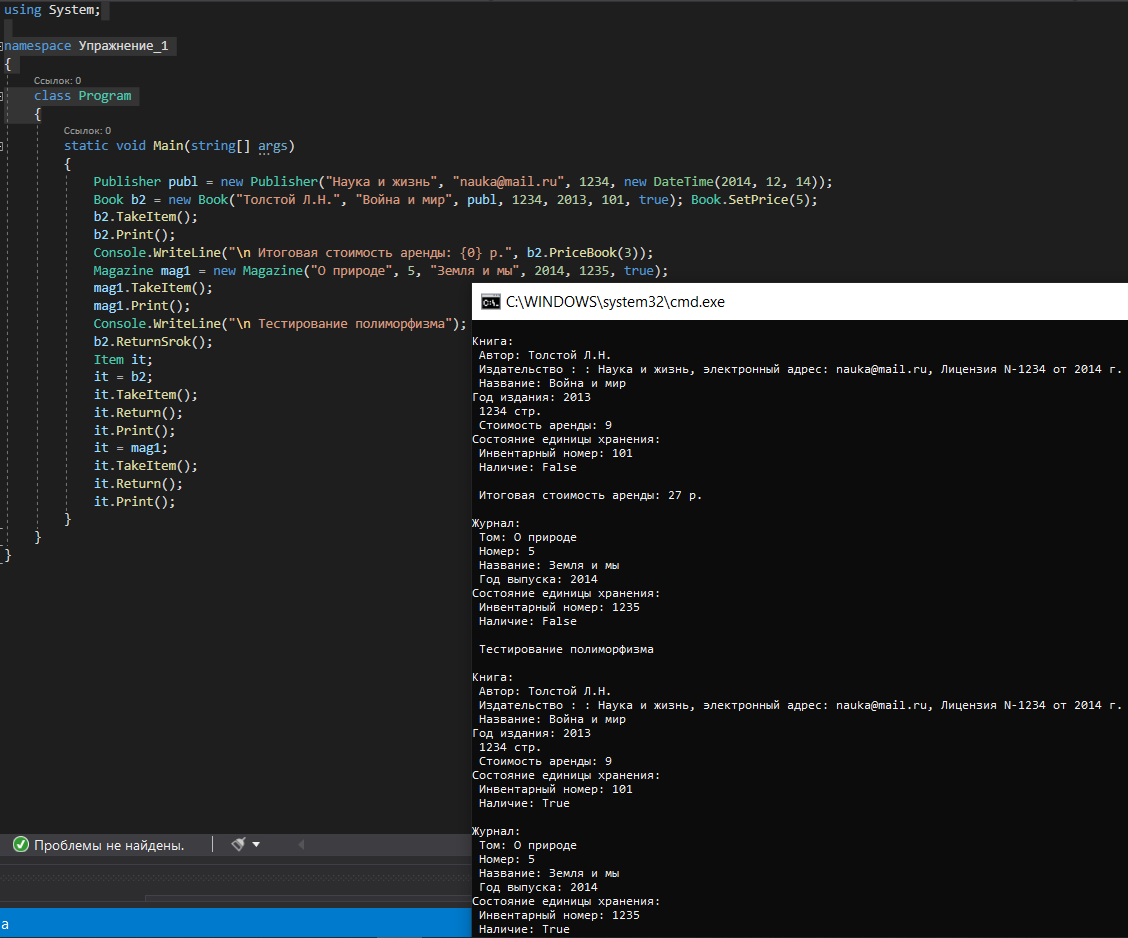


Рисунок 17 – Конечный результат работы программы

**Упражнение 7.** Реализация прогрессии.

Создан абстрактный класс Progression, с одним абстрактным методом GetElement, принимающим один параметр - номер элемента, который нужно найти. Определены два производных класса ArithmeticProgression и GeometricProgression, описывающие арифметическую и геометрическую прогрессии. В каждом из них определён конструктор, задающий параметры прогрессии и переопределён унаследованный метод GetElement. В методе Main создаются объекты производных классов с помощью конструкторов и выводятся на консоль значения элементов, указанных пользователем.

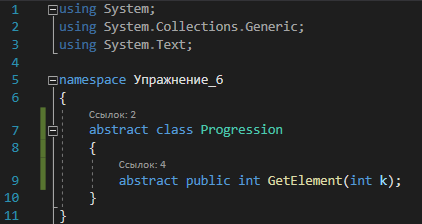


Рисунок 18 – Класс Progression

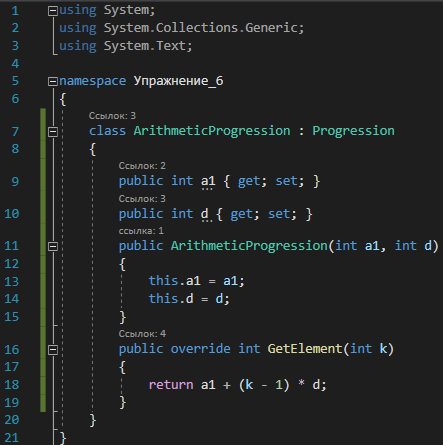


Рисунок 19 – Класс ArithmeticProgression

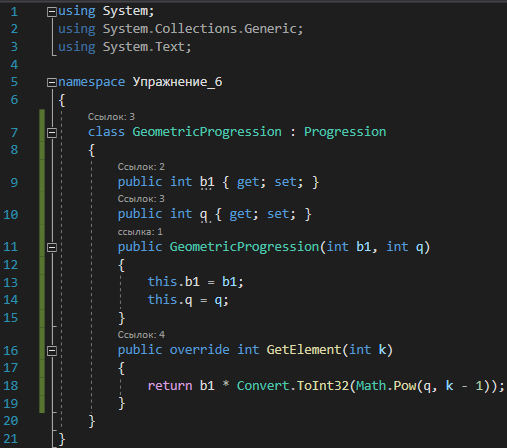


Рисунок 20 – Класс GeomerticProgression

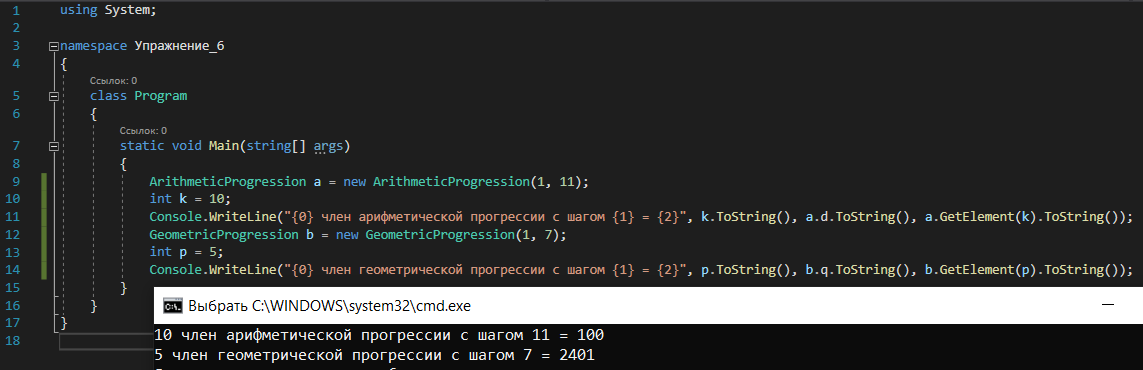


Рисунок 21 – Основная логика и результат работы программы

**Вывод:**

В результате выполнения лабораторной работы рассмотрено: отношения наследования классов; наследование базовых конструкторов и базовых методов с помощью ключевого слова base; переопределение методов с помощью ключевого слова override; полиморфный вызов функций производных классов, путём указания ключевого слова virtual перед методом родительского класса, полиморфный вызов функций позволяет использовать один интерфейс для выполнения однотипных, но отличающихся задач, что позволяет снижать сложность программ; создание абстрактных классов и методов, выполняющих роль шаблона, от которых нельзя создавать экземпляры классов; различные отношения между классами: ассоциации, включения, агрегации; различные модификаторы доступа: public, protected, private.