# Exercise: Inheritance and Abstraction

This document defines an in-class exercise from the ["OOP" Course @ Software University](https://softuni.bg/courses/oop/).

## Book Shop

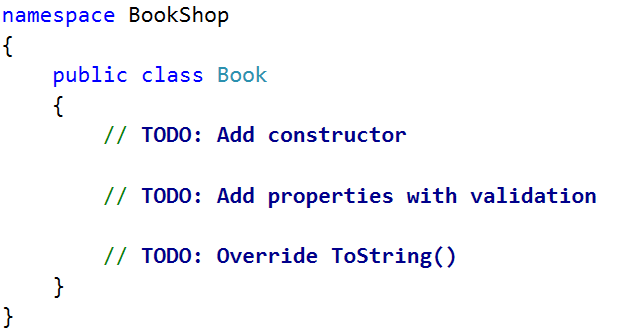
Our program will work with the following classes:

* **Book** - represents a book that holds **title**, **author** and **price**. Validate that the title and author are not null. The price should never be a negative number. A book should offer **information** about itself in the format shown in the output below.
* **GoldenEditionBook** - represents a special book holds the same properties as any **Book**, but its **price** is always **30% higher**.

|  |  |
| --- | --- |
| **Sample Code** | **Output** |
|  | -Type: Book  -Title: Pod Igoto  -Author: Ivan Vazov  -Price: 15.90  -Type: GoldenEditionBook  -Title: Tutun  -Author: Dimitar Dimov  -Price: 29.77 |

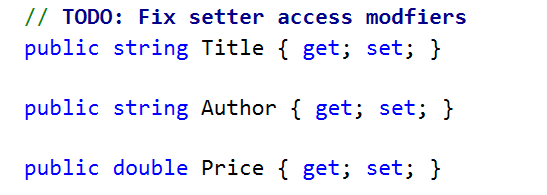
### Step 1 - Create a Book Class

Create a new empty class a name it **Book**. Set its access modifier to **public** so it can be instantiated from any project.



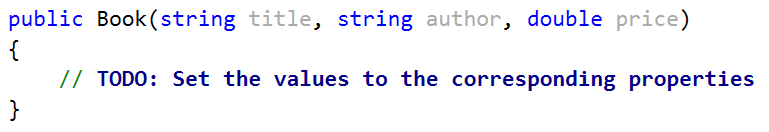
### Step 2 - Define the Properties of a Book

Define the **Title**, **Author** and **Price** properties of a Book. Ensure that they can only be **changed by the class itself or its descendants** (pick the most appropriate access modifier).



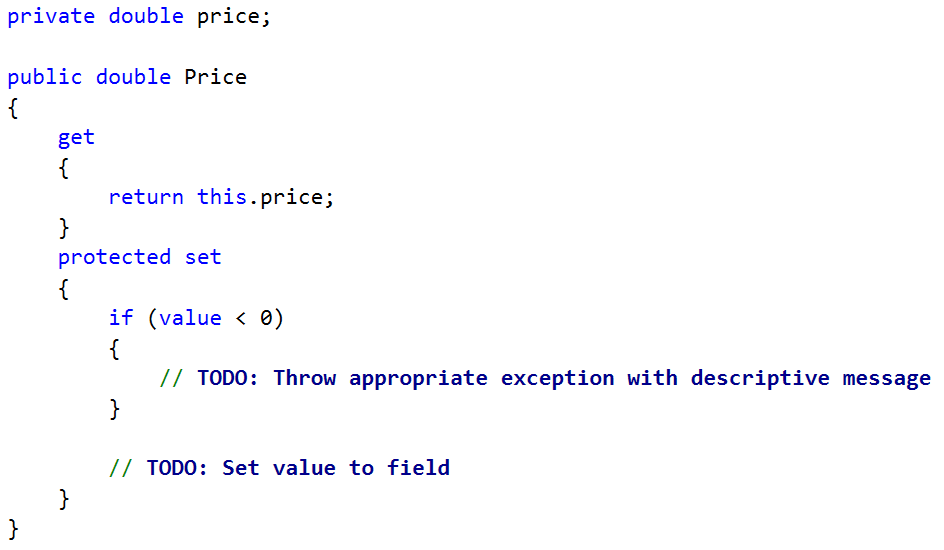
### Step 3 - Define a Constructor

Define a constructor that accepts **title**, **author** and **price** arguments.



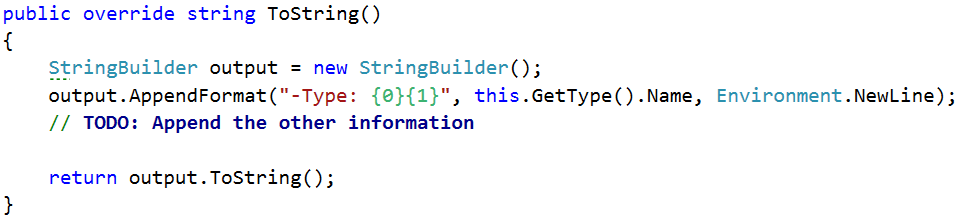
### Step 4 - Perform Validations

Create a **field** for each property (**Price**, **Title** and **Author**) and **perform validations** for each one. The **getter should return the corresponding field** and the **setter should validate** the input data before setting it. Do this for every property.



### Step 5 - Override ToString()

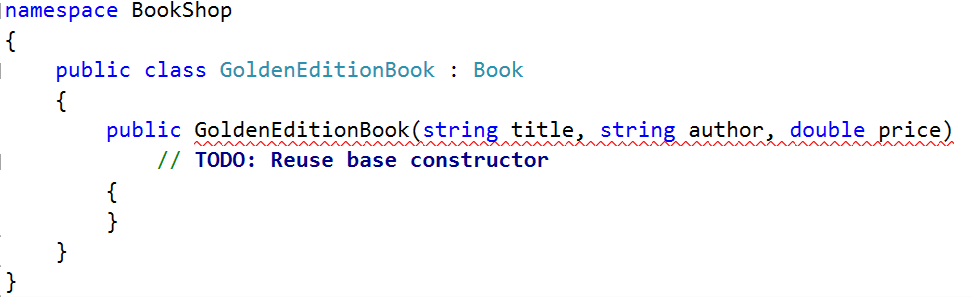
As you probably already know, all classes in C# inherit the **System.Object** class and therefore have all its **public** members (**ToString()**, **Equals()** and **GetHashCode()** methods). **ToString()** serves to return information about an instance as string and is declared **virtual** (therefore can be changed in descendant classes). Let's **override** (change) its behavior for our **Book** class.



And voila! If everything is correct, we can now create **Book objects** and display information about them.

### Step 6 - Create a GoldenEditionBook

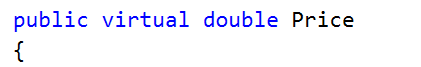
Create a **GoldenEditionBook** class that inherits **Book** and has the same constructor definition. However, do not copy the code from the Book class - **reuse the Book class constructor**.



There is **no need** to rewrite the Price, Title and Author properties since **GoldenEditionBook** inherits **Book** and by default has them.

### Step 7 - Override the Price Property

Golden edition books should return a **30%** higher **price** than the original price. In order for the getter to return a different value, we need to override the Price property. To do that, go back to the **Book** class and declare it **virtual**.



**Virtual properties/methods** can be **overridden** in descendant classes.

Back to the **GoldenEditionBook** class, let's override the Price property and change the getter body.

