CIT 243
Assignment 02

The public library in town has a system that is used to search for items, check them out, return them, and renew them. For this assignment, we will create a simplified model of this system, using OOP and inheritance. In Part 1 and Part 2, we will create a hierarchy of classes whose objects will represent items that can be checked out at the library. In Part 3 we will create an application that uses objects of these classes.

Part 1: the base class

Create the `LibraryLoanItem` class that represents a generic item that the library lends out to its patrons. This class will have many of the characteristics of every item in the library, such as, a call number, title, author, etc., as described below.

Variables:

1. `callNumber` – consists of a 3-digit number followed by a space, and the first three letters of the author's last name in upper case, e.g., the call number for our text book could be 100 GAD.
2. `title` – the title of the item
3. `author` – the author of the item
4. `copies` – the number of copies of this item the library has
5. `availableCopies` – keeps a count of the number of copies that are currently available for check out
6. `loanPeriod` – how many days the item can be checked out for
7. `maximumRenewals` – the maximum number of times this item can be renewed.
8. `timesRenewed` – an array that contains counts of how many times each copy of this item has been renewed.

Remember to declare your variables as `protected`, so that derived classes can inherit and directly access them.

Constructor:

1. Has the call number, title, author, loan period, number of copies and maximum renewals as parameters.
2. Checks all parameters to make sure they are valid. If so, initializes the corresponding variables. Maximum renewals can be zero or more.
3. Initializes `availableCopies`. Initially all copies are available.
4. Initially, none of the copies has been renewed.

Properties:

Provide read-only properties for all of the variables.

Methods:

1. `CheckOut` – if at least one copy of this item is available, checks it out, and returns the copy number, or -1 if no copies are available.
2. `CheckIn` – checks in a specific copy of this item which had been borrowed
3. `Renew` – can be called to renew a specific copy of this item; is a `virtual` method.
   Both of the above methods return true/false to indicate success or failure.
4. `ToString` – returns a String representation of this item (include call number, title, author, loan period, number of copies, and available copies in the `String` representation)

Part 2: the derived classes

Create the following classes that are derived from `LibraryLoanItem`. Make sure you provide a constructor for each one, and that the constructor first calls the base class's constructor and then performs any special initialization each class requires.

1. `Book` – can renew a maximum of 4 times. Loan period = 21 days.
2. `CD` – can renew 2 times. Loan period = 21 days.
3. `Video` – can renew once. Loan period = 7 days.
4. `NewBook` – is a derived class of `Book`. A `NewBook` is not renewable. Loan period = 21 days.
5. `DVD` – is not renewable. Loan period = 7 days.

In each derived class,

1. override `ToString` so that it includes the type of the item in addition to what the base class's `ToString` method returns.
2. In the ones that are not renewable, override `Renew` so that it simply returns `false`.

Part 3: the application

Create an application where you allow the user to:

1. Add an item to the library's collection
2. Check out an item by providing the call number
3. Check in an item by providing the call number and copy number
4. Renew an item by providing the call number and copy number
5. Search by title
6. Search by author

When any of these operations is done, show a confirmation and display the item's call number, title, author, loan period, number of copies, and available copies, or a message that the operation failed or that the item could not be found.

Submit your zip file on Blackboard.