PCS3 Exam

Date: Wednesday, 2018, April 4th
Time: 12:45 – 15:15 h. (150 minutes)

Rules:

Admitted resources:

- You are allowed to use everything on paper (books, notes, etc.) and on your laptop, but only what you bring in: you are not allowed to borrow something from someone else.
- During the exam it is not allowed to use the network. You should make the exam yourself: so, no communication with msdn or google for help and no communication with other students via Facebook, e-mail, Skype, gsm or whatever.

Way of working during the exam and handing in your exam:
During the exam you will use a program that blocks your internet connection, takes care of downloading the exam (this description and a startup-project) and let you upload your solution. If needed, you can read a user's guide which is present at the exam.
Preliminary remarks:

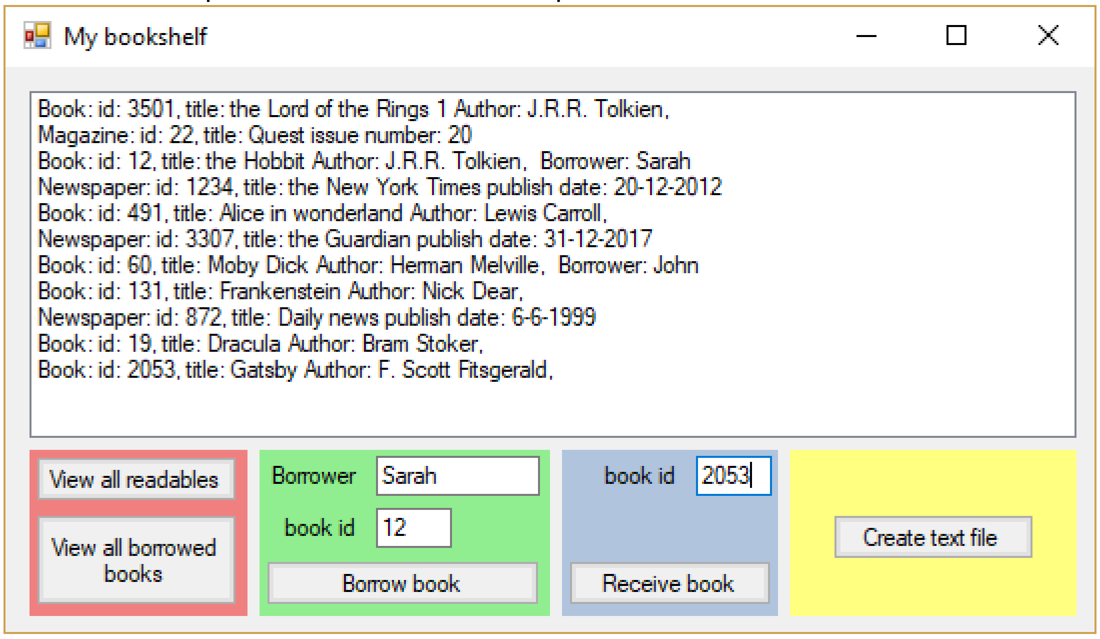
- Whenever this exam paper suggests to use a certain name for a method, variable or anything else, you are required to indeed use that name.
- If you think you need more class-members, you are allowed to add more members to the classes than specified in this exam.

Introduction: An application to manage a bookshelf.

One of your friends loves to read. He has a big bookshelf filled with books, magazines and newspapers, let's call them *"readables"* from now on. He has asked you to create an application that has the following functionality:
The user should be able to see which readables he/she owns. It should be possible for friends to borrow the user's books. Newspapers and magazines are too fragile, so it should not be possible to borrow them. For this reason the application should register which books are borrowed and who currently borrows them. Last, it should be possible to create a text file that contains information about the books he borrowed (more details will follow in the relevant assignment).

In this exam you will implement part of the C# application that will provide these functionalities.

The screendump below demonstrates the provided GUI.



```
Book: id: 3501, title: the Lord of the Rings 1 Author: J.R.R. Tolkien,
Magazine: id: 22, title: Quest issue number: 20
Book: id: 12, title: the Hobbit Author: J.R.R. Tolkien,  Borrower: Sarah
Newspaper: id: 1234, title: the New York Times publish date: 20-12-2012
Book: id: 491, title: Alice in wonderland Author: Lewis Carroll,
Newspaper: id: 3307, title: the Guardian publish date: 31-12-2017
Book: id: 60, title: Moby Dick Author: Herman Melville,  Borrower: John
Book: id: 131, title: Frankenstein Author: Nick Dear,
Newspaper: id: 872, title: Daily news publish date: 6-6-1999
Book: id: 19, title: Dracula Author: Bram Stoker,
Book: id: 2053, title: Gatsby Author: F. Scott Fitsgerald,
```

Borrower: Sarah

book id: 12

book id: 2053

In the screendump above you see the application after a certain amount of user interaction. At the top you find a listbox (lbReadables) which displays the owned readables. Below you will find four panels from left to right:

- (left) Red: This panel contains two buttons: “View all readables” and “View all borrowed books”. They will display readables in the listbox according to the button text.
- Green: This panel contains the controls related to borrowing a book to a borrower. The name of the borrower and the id of the book are required for this. (e.g.: we have provided “Sarah” as borrower, and “12” as book id)
- Blue: This panel contains the controls related to receiving the book back from the borrower. The id of the book is required for this. (e.g.: we have provided “2053” as book id)
- (Right) Yellow: This panel contains one button. This button should create the text file explained below.

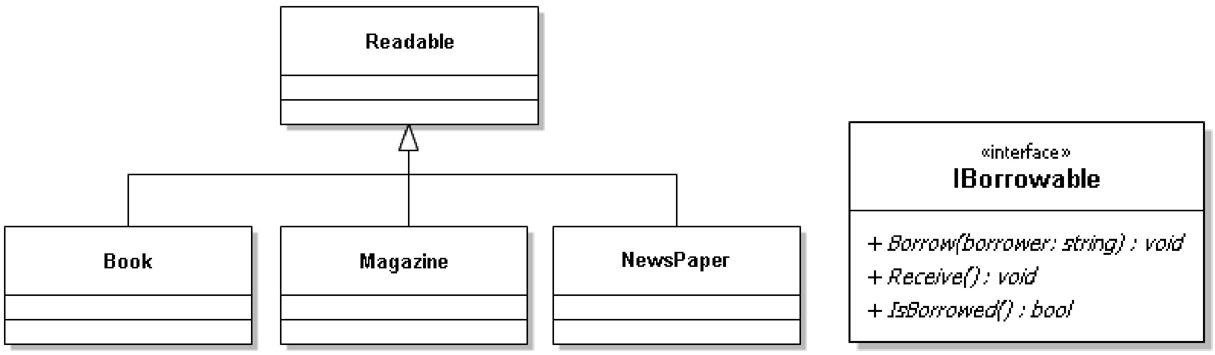
To help you test your application during the exam, the method “GenerateDummyReadables” is provided. This method creates readables.

<u>The startup project contains some code already, but it does not compile yet.</u>

Assignment 1: Readables and Borrowables (40) pts

In this assignment we ask you to create the classes that represent the different readables. Below you will find a diagram which depicts the relation between the different classes that you will provide. A more detailed description will follow.

In this assignment we expect you to use the benefits of inheritance as much as possible.



There are three readables: **Book**s, **Magazine**s and **NewsPaper**s. Additionally you are provided with an interface **IBorrowable**. This interface describes what methods should be implemented by a class that allows the user to borrow it to others. You are not allowed to change the interface **IBorrowable**.

Add the classes to your project and provide code according to the following rules:

1. Every readable has a unique identity number (int).
   Every readable has a title.
2. Every readable contains a method that returns its information as a string: first a text indicating its type and then information of its data, like in the listbox seen in the screendump above.
3. In the method GenerateDummyReadables in MyBookShelfForm you can find the format of the constructors for each readable. All constructors start with the parameter for the id and the title.
4. The class **NewsPaper** registers the date (string) on which it was published.
5. The class **Magazine** registers the issuenumber of the magazine.
6. The class **Book** must implement the interface **IBorrowable**. It should register the author, the current borrower and the number of previous borrowers.
   The method **IsBorrowed** should inform if the book is borrowed or not. If the book is borrowed, it will return true, if the book is not borrowed, it will return false.
   The method **Borrow** should check if the book is not currently borrowed. In case the book is not borrowed, this method will change the borrower to the provided name. In the case that it is borrowed, it will not change the current borrower.
   The method **Receive** should register that a borrowed book is received by the owner. So when the book is borrowed, this method changes the borrower to the empty string and increases the number of previous borrowers by 1. If the book is not borrowed, nothing happens.

Assignment 2: The Bookshelf (10 + 5 + 5 + 10) pts

As you may have noticed, the **Bookshelf** class is already partially implemented. You are provided with the code that: stores the readables, adds them, gets all of them and gets a specific readable. You are not allowed to change this code.

a. Borrowing a book to someone:

Create the following method: **void BorrowReadable(int id, string name)**.
This method should make it possible to borrow a readable to a borrower. In case the readable is borrowable and not yet borrowed it will return true. When the readable is borrowable and borrowed, the method should return false.
However, not every readable can be borrowed. Whenever the user attempts to borrow a readable which is not allowed to be borrowed, you should throw your own exception “NotABorrowableException”. The message should make it clear why this cannot happen.

b. Receiving a book from someone:

Create the following method: **void ReceiveReadable(int id)**.
This method should make it possible to receive a readable from a borrower. Similarly to borrowing, we do not want a user to borrow readables that cannot be borrowed. To prevent this, you should throw the “NotABorrowableException” when the user attempts this. The message should make it clear why this cannot happen.

c. Getting all borrowed books:

Create the following method: **List<Readable> GetAllBorrowedReadables()**.
This method should return a list of all books that are borrowed.

d. Create the text file:

Create the following method: **void GenerateOverviewInFile(string filename)**.
This method will create a text file at the provided filename. The file is built up in the following way:

The file starts with the text: “Book Report:”. The next line provides the number of owned books. This is followed by the number of currently borrowed books. Last, separated by a line of asterisks (*) for every owned book, the title and number of previous borrowers is provided.

Keep in mind that the user will eventually own (purchase) more books.

The example below displays a textfile resulting from the example in the screendump above:

```
Book Report:
You currently own 7 books.
You have currently borrowed 2 books.
*******************************
the Lord of the Rings 1 is borrowed 3 times.
the Hobbit is borrowed 5 times. Sarah currently borrows this book.
Alice in wonderland is borrowed 20 times.
Moby Dick is borrowed 1 times. John currently borrows this book.
Frankenstein is borrowed 2 times.
Dracula is borrowed 13 times.
Gatsby is borrowed 0 times.
```

Assignment 3: the GUI (2 + 3 + 8 + 8 + 9) pts

Last but not least you are asked to implement code in the GUI. Just like the previous assignment, some code has been provided. You are not allowed to change this code.

a. Displaying all readables:

Clicking the button with text “View all readables” should display all readables that are owned by the user in lbReadables. Make it work.

b. Displaying all borrowed books:

Clicking the button with text “View all borrowed books” should display all books that are currently borrowed in lbReadables. Make it work.

c. Borrowing a book:

First the user has to provide the name of the borrower and the book id.
By clicking the button with text “Borrow book”, one of two things can happen:

- The readable can be borrowed: In this case the readable is borrowed.
- The readable cannot be borrowed: The user is informed that this readable cannot be borrowed. You are free to choose how to display this (MessageBox, label, etc.).

Handle all kinds of possible exceptions.

d. Receiving a book:

First the user has to provide the book id.
By clicking the button with text “Receive book”, one of two things can happen:

- The readable can be returned: In this case the readable is returned.
- The readable cannot be returned: The user is informed that this readable cannot be borrowed. You are free to choose how to display this (MessageBox, label, etc.).
  Handle all kinds of possible exceptions.

e. Create a text file about the books:

Clicking the button with text “Create text file” should open a save file dialog. When the user presses the OK button, a book overview according to the format of assignment 2d should be saved to the specified file.

----- END OF THE EXAM