Multiply-Linked Lists

(aka a 'threaded' linked list)

BIT 143 – ASSIGNMENT 3

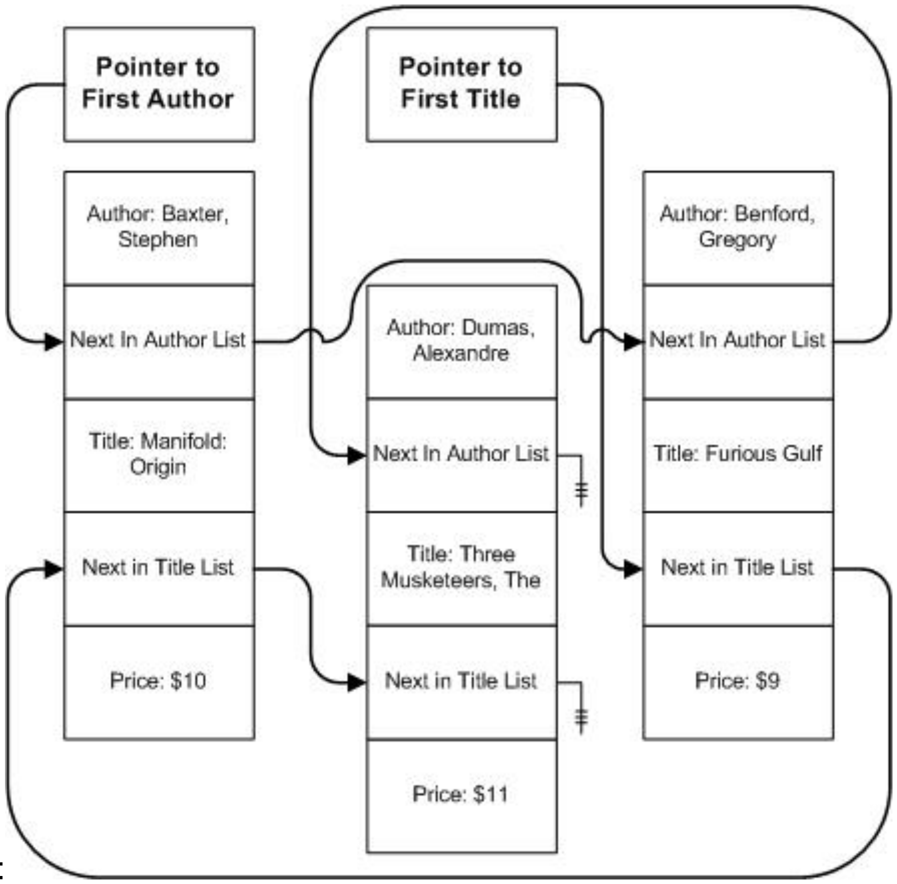
Due: < Listed in the course schedule >

You are not allowed to work in groups this assignment. For this assignment, you should start, finish, and do all the work on your own. If you have questions, please contact the instructor.

What the program must accomplish

Imagine that you're going to keep track of some data. In this case, you're going to keep track of a collection of books. Each book has an author (a string), a title (another string), and a price.

For this assignment, you're going to create a program that will manage a linked list of books. There's a twist, however – each node in the linked list will actually have TWO next pointers, so that you can have two 'lists' stored in the same place. There's going to be 1 list for the books, sorted according to the author's name, and there's going to be a second list, sorted according to the book's title. A picture of how this might look is given below:



:

Notice how each Book has a 'next author' AND a 'next title' reference.
Thus, if we were to print the books by following the links in the author list, we'd get:

```
Book Title: Manifold: Origin
Book Author: Baxter, Stephen
Price: $10

Book Title: Furious Gulf
Book Author: Benford, Gregoru
Price: $9

Book Title: Three Musketeers, The
Book Author: Dumas, Alexandre
Price: $11
```

Likewise, if we were to print the books by following the links in the title list, we'd get

```
Book Title: Furious Gulf
Book Author: Benford, Gregory
Price: $9

Book Title: Manifold: Origin
Book Author: Baxter, Stephen
```

```
Price: $10

Book Title: Three Musketeers, The
Book Author: Dumas, Alexandre
Price: $11
```

You need to implement a program that will allow someone to add a book, remove a book, and print the books by either author or title. The Book objects must be stored in the 'multiply linked lists' described above. If you have two books with the same author, then they should be put into sorted based on their different titles (and likewise with books with the same title, but different authors). If you've already got a book with a particular author AND title, and you try to add another book with exact same author & title, you can indicate that an error has occurred, and not add the (duplicate) book.

To be clear: you need to implement not just the data structures that store this information, but you also need to finish implementing the Console-based user interface that will allow the user to interactively create books, print the list, and remove books that is partially provided to you in the starter project. There are a couple of comments that start with `// STUDENTS:` that should identify areas that you need to complete (in addition to the multiply linked list work). Make sure that you fill in any error-handling code in the UserInterface class that needs to be finished, as well as any unfinished methods!

Just as an FYI, I don't think that the term 'multiply linked' is a standard term. DOUBLY LINKED lists are a standard term (in a doubly-linked list is one in which each node has both a next pointer, and a **previous** pointer – a link to the node prior to it in the list), but what you'll be doing here is different from a doubly linked list.

Commenting:

You should comment your code, paying particular attention to areas that are difficult to understand. If you found something to be tricky when you wrote it, make sure to comment it so that the next person (the instructor, who's grading you) understands what your code is doing. It is not necessary to comment every single line.

The purpose of this requirement is to both help you understand, and have you demonstrate, a thorough understanding of exactly how your program works.

Every file that you turn in should have:

- At the top of the file, you should put your name (first and last), the name of this class ("BIT 143"), and the year and quarter, and the assignment number, including the revision number, which starts at 0 ("A3.0"). If you're handing this in again for a regrade, make sure to increase the minor version number by one (from "A3.0", to "A3.1").

In general, you should make sure to do the following before handing in your project:

- All variables used should have meaningful names.
- The code should be formatted consistently, and in an easy to read format.

<u>What to turn in:</u>

- A single electronic folder (a directory). This folder should contain:
    - The C# source code for the entire program. This will be all the .CS files that Visual Studio.Net creates for you.
    - Any other files that Visual Studio.Net has created for you, including the solution file (.SLN), and the project file (.CSPROJ). Basically, you need to hand in the entire directory that VS.Net created for you. You should leave out subfolders (such as the **Debug, bin**, or **obj** directories) that are generated from the source code/project.

<u>How to electronically submit your homework:</u>

There's a link on the homework page to the document that guides you through handing in your work.