PROGRAM 2: Catalog

Due
The due date is specified in the OAKS dropbox.

OBJECTIVES

- To continue to learn Eclipse.
- To use an interface to enforce the ADT definition within the implementation.
- Become familiar with implementing a singly-linked linear list class.
- Practice with Java file and console I/O.
- To open a read an XML file.
- To use a linear, abstract data type, List, implemented as a linked list yourself using Node objects.
- To test the application.

PROGRAM REQUIREMENTS
Write a program to read a data file from an XML file into memory. Specifically, the data file is an XML file containing a catalog of book data.

Input:
De-serialize the XML data by reading each book into a Book object and adding that book object to the Catalog object, a singly linked list of books. An input file is provided for development. The program will output the following information to the console.

Output:
A menu driven system provided by the Catalog object will allow the user to do the following:

1. **Add** a book to the catalog. (allows for duplicate book entries)
2. **Remove** a book from the catalog by title. (exact title string match)
3. **Search** for all books. (search by a given sub-string in the title)
4. **Print** the catalog. (onto the console, all at one time)
5. **End** program. (automatically saves the catalog to a file for next time)

Option: <user types in an integer here>

Label and format the output to the console. Your formatting will be graded relative to your classmate’s job at text formatting in the console window.

REFERENCES
none

PROGRAM SPECIFICATION
Application: All classes reside in a package called **library**. Call the class with the main method, **RunCatalog.java**. The main program is expected to interact with the **Catalog** object to produce the results on the console. Create other helper classes as you need them to create a good object oriented design paying attention to OO design principles.

Abstract Data Type: Implement a List ADT using dynamic memory allocation (a Linked List implementation). For best results, do not copy a linked list implementation from the Web. Implement your own. You will need to first implement the Node class that will be used by the SinglyLinkedList class.

Create a generic **SinglyLinkedList** class to hold any type of object. This is the class that implements the List ADT (by implementing the generic **ListInterface**) and that Catalog uses.

Create a **Book** class with private attribute names taken from the XML data file (or from the book XSD).

Create a **Catalog** class that uses a singly linked list to store and manipulate book objects as needed by the program requirements.

API: None from Java Collections is needed at this time.

GUI: None required.

Design Constraints: Use String parsing to read elements from the file to populate the Book objects. Be patient. At a later time, we can rewrite the program using a DOM parser from java.xml. (The point of the program is to work with a list and not a tree.)

So I am first asking you to implement your own way to read this specific XML file, that is the specific xsd – XML Schema Definition – that underlies this XML documents (instance of the XSD).

PROGRAM DOCUMENTATION

Provide internal documentation only as required in the program documentation standard in OAKS.
Updates and clarifications to this assignment, if needed, will be done on Discussions.

PROGRAM SUBMISSION

Use the corresponding dropbox in OAKS using the same naming conventions as given in Program 1.