CIS351

Homework 5

Orange Library

Objectives

- Implement a specific design from UML diagrams.
- Validate the arguments before updating an object.
- Write toString methods that use string formatting.
- Search for given values within an array of objects.
- Solve problems using object-oriented techniques.

Download Materials

Download the following files from Blackboard

- DateUtils.javaPreview the document - This utility class is provided for you.
- Driver.javaPreview the document - This example class is provided for you.
- Library.javaPreview the document - The Library class is provided for you.
- lflbooks.txtPreview the document - Example books.
- patrons.txt - Example patrons.

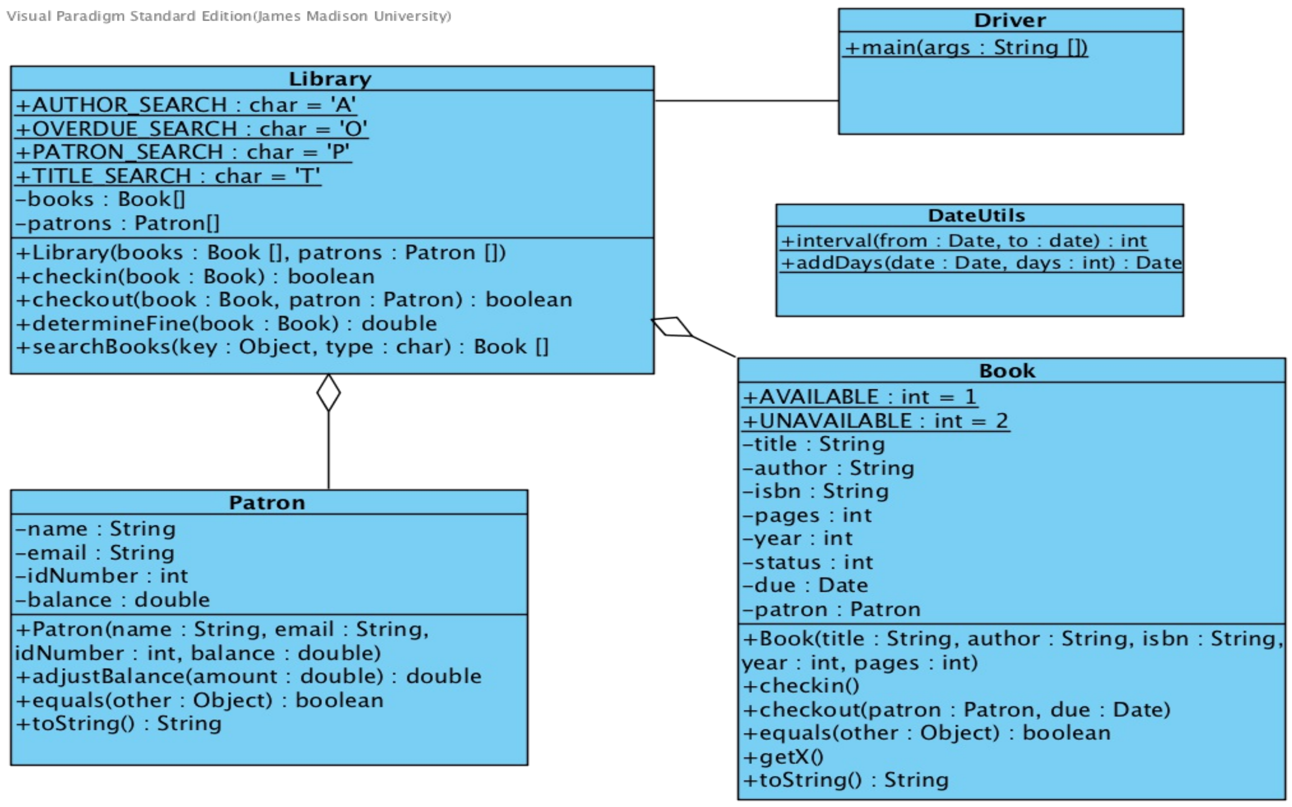
Background

This assignment is inspired by the Little Free Library organization which has managed to charter small "libraries" like the one shown above all across the world. Each "library" encourages patrons to take a book or leave a book to promote the accessibility of books and reading in general. They are community spaces, many with benches or seats nearby and all are maintained by volunteers. There are eight in Harrisonburg, Virginia. The data file you will be using in the assignment is based on the books available at this writing in the Elizabeth Street box.

Requirements

In this assignment, imagine that the Little Free Library organization has contracted with the SU EECS department to track how much money they could earn if they were a commercial organization that charged late fees for overdue books. This information will then be used for local librarians to apply for funding from other non-profit foundations or the government. Each book and patron will be modeled by classes with relevant data and methods as outlined in the UML diagram and descriptions shown below. The Library class will then have arrays of books and patrons along with methods that allow for books to be checked in, checked out, and found in addition to determining fines.

Look at the following UML Diagram of this library system. All of the required files are provided for you except **Patron.java and Book.java**, which you need to implement in this homework.



Visual Paradigm Standard Edition(James Madison University)
Driver
+main(args : String [])
Library
+AUTHOR_SEARCH : char = 'A'
+OVERDUE_SEARCH : char = 'O'
+PATRON_SEARCH : char = 'P'
+TITLE_SEARCH : char = 'T'
-books : Book[]
-patrons : Patron[]
+Library(books : Book [], patrons : Patron [])
+checkin(book : Book) : boolean
+checkout(book : Book, patron : Patron) : boolean
+determineFine(book : Book) : double
+searchBooks(key : Object, type : char) : Book []
DateUtils
+interval(from : Date, to : date) : int
+addDays(date : Date, days : int) : Date
Patron
-name : String
-email : String
-idNumber : int
-balance : double
+Patron(name : String, email : String, idNumber : int, balance : double)
+adjustBalance(amount : double) : double
+equals(other : Object) : boolean
+toString() : String
Book
+AVAILABLE : int = 1
+UNAVAILABLE : int = 2
-title : String
-author : String
-isbn : String
-pages : int
-year : int
-status : int
-due : Date
-patron : Patron
+Book(title : String, author : String, isbn : String, year : int, pages : int)
+checkin()
+checkout(patron : Patron, due : Date)
+equals(other : Object) : boolean
+getX()
+toString() : String

You must read through the code of **Library** class to understand the structure and interaction in this program. You don't actually need the Driver to complete the assignment; it is simply provided to show how your classes might be used in a larger application.

HELP: DO NOT MODIFY Library.java. It is provided for you. Please read this class very carefully to understand what each method is doing. This will also help you to code *Patron.java* and *Book.java*.

Specification of classes that YOU NEED TO CODE

Book.java (YOU NEED TO CODE)

1. The Book `constructor` will be used to initialize all attributes of the Book object. The initial status of the book should be AVAILABLE, and the patron and due date should be null.

2. Like the constructor, the `checkin` method should set the book's status to AVAILABLE and assign null to both patron and due date.

3. The `checkout` method should make the book UNAVAILABLE and save the given patron and due date in the object.

4. The `equals` method checks whether two books have the same isbn. If the given object is a Book, compare this.isbn with the other book's isbn. If the given object is a String, compare this.isbn with the string.

5. In the UML diagram, `getX` is actually eight methods: one accessir method for each of the eight attributes. These should be easy to implement (Eclipse will generate them automatically). Note that the class should have no "setX" methods.

6. The `toString` method should return a String representation of the book. If a book is initialized with the following parameters `("aaa", "bbb", "1234BBBBBBBBB", 2013, 10)`, it should return a String *exactly* as follows: `"Title: aaa, Author: bbb, ISBN: 1234BBBBBBBBB, Year: 2013, Pages: 10."`

Patron.java (YOU NEED TO CODE)

1. The Patron `constructor` will be used to initialize all attributes of the Patron object.

2. The `adjustBalance` will update the current balance of the patron. For example if the current balance of a Patron is 2.0 and the amount is 4.5, this method should assign 6.50 to this.balance and return 6.50.

3. The `equals` method checks whether two patrons have the same id number. If the given object is a Patron, compare this.idNumber with the other patron's idNumber. If the given object is an Integer, compare this.idNumber with the Integer.

4. The `toString` method should return a String representation of the Patron. If a patron is initialized with the following parameters ("bob", "eee", 2, 5.5), it should return a String *exactly*

as follows: "Name: bob, Email: eee, ID: 2, Balance: $5.50."

Hints and Tips

- Do not implement any other methods than what you see in the UML diagram. For example, you should not have a Patron.getBalance() method, even for testing purposes.

- JUnit's assertEquals method automatically calls the object's equals method. So instead of writing assertTrue(book1.equals(book2)), you should just write assertEquals(book1, book2).

Submission Instruction

1. Submit completed Book.java and Patron.java. See the submission process here
2. Submit a filled out version of the standard **cover sheet**.

Grading Criteria

Total points: 100 points

Correctness of Patron.java - 50pt
Correctness of Book.java - 50pt

Don't forget
the submission process or
the grading criteria or
the cover sheet.

Farzana Rahman / frahman@syr.edu