Assignment - MiniLibrary Basic Operations + Book Reservation

[Weight: 12 marks out of the final mark of this course]

Deadline: December 02, 2016 (Friday of revision week)

For late submissions, 2% of your original marks will be deducted if you hand in 1-day late (i.e. on Dec 03), 4% for 2-days (i.e. on Dec 04), 8% for 3-days (i.e. on Dec 05), 16% for 4-days (i.e. on Dec 06), 32%for 5-days (i.e. on Dec 07), 64% for 6-days (i.e. on Dec 08). Assignments handed on or after Dec 09 get 0 mark.

Academic dishonesty is strictly prohibited. The principle concerns whether students get their deserved marks and do not intend to cause unfairness. Dishonesty also involves when one let others have a chance to copy his/her code,

Grading:

Students **must** obtain the following results in sequence:

Phase 1 (100% correct in PASS) ==> Phase 2 (100% correct in PASS) ==> Phase 3(100% correct in PASS)

- If you can finish Phase 1 with good programming styles + OO programming skills => up to B
- If you can finish Phase 2 with good programming styles + OO programming skills => up to A-
- If you can finish Phase 3 with good programming styles + OO programming skills => up to A+

Various test cases are used for each phase (eg. Phase 1: 1a.txt, 1b.txt, etc..). If you get partial correct, your work is still considered. Eg. If you can pass 1a.txt only, your grade may be up to C.

For "Good Programming Styles", note that proper indentations , code-layout formatting, proper , meaningful naming, well-designed classes, methods, fields are more important than writing comments.

Assignment Description

Note: Before you start working on the assignment, please first learn from Lab09-Q2 mini-library and Lab10 Section I

Extend the `Lab09-Q2 mini-library` application to allow adding library books, borrow and return books, reserve books (i.e., queue up for a book which is unavailable at the moment) and pick up books.

1. Revise the existing classes and add new classes for proper modelling of the system.
   - Add an array list of Book objects in the Library class.
   - Add the Book class (object fields involve book ID, name, arrival date, and book status etc..)
   - Reservation and queue management
     [Modeling] There are a number of ways to model the reservation queue. One approach is to keep a queue (ArrayList) of reservations for each book. You may use any other design.

     [Logistic] When a book is returned and the request queue is not empty, the first requesting member will be removed from the request queue and the book is marked as onhold for this member to pick up. The onhold period will due in 3 days. If this member does not pick up the book within the period, the next queuing member will take turn after it is due.
   - The book status (available or borrowed etc.) should be implemented using the State Pattern (Learnt in Lab06):
     - public interface BookStatus
     - public class BookStatusAvailable implements BookStatus
     - public class BookStatusBorrowed implements BookStatus
     - public class BookStatusOnhold implements BookStatus

     BookStatusBorrowed etc. may contain fields like: the borrowing member, loan date
   - The Member objects should keep track of borrow counts and request counts, which are shown upon `listMembers`.

2. New commands for the library operations involve
   i. `arrive` : arrival of new books
   ii. `checkout` : a member borrows at most 6 book (loan period is not catered in this assignment)
   iii. `checkin`: a member returns a book.
   Note 1: See P. 1 for "[Logistic]" in point 1;
   Note 2: Undo/redo can be complicated if there is any queuing member.
   iv. `listBooks`: listing of all books in the library
   v. `request`: a member requests to queue up for an unavailable book. A member can have at most 3 active book requests at the same time. Note: a member is not allowed to request an available book, or request a book which this member is currently borrowing, or is already queuing for it.
   vi. `cancelRequest`: a member cancels a request. Note: undo/redo can happen!

3. The commands `startNewDay, listMembers` and `register` were started in `Lab09-Q2` already. You may need to further modify some of them.

   Note for `startNewDay`:

   Cases of "onhold due" are to be handled upon `startNewDay`. (See P. 1 for "[Logistic]" in point 1.)
   However, for simplicity your program does not need to handle undo/redo of StartNewDay.

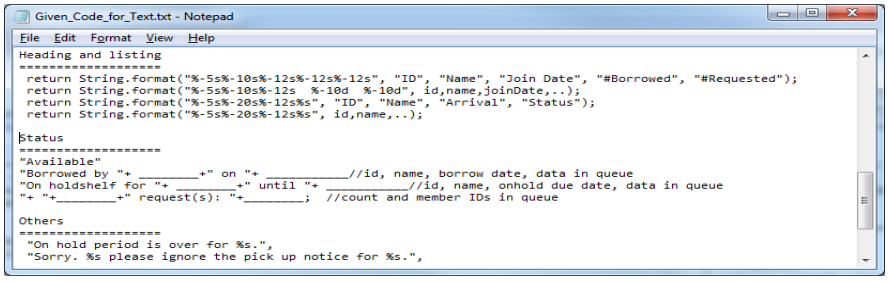
4. The names of command classes should start with "`Cmd`", eg. "`class CmdRegister`", "`class CmdListBooks`"

5. You will need to add handling for the following error cases
   a) Member ID already in use (For register)
   b) Book ID already in use (For book arrival)
   c) Member not found (For checkout)
   d) Book not found (For checkout)
   e) Book not available -- already borrowed by or on hold for somebody (For checkout)
   f) Loan quota Exceeded (For checkout)
   g) The book is not borrowed by this member (For checkin)
   h) The book is currently available (For request -- if it is available, or on hold for the requesting member to pick up)
   i) The book is already borrowed by the same member (For request)
   j) The same member has already requested the book (For request)
   k) Book request quota exceeded (For request)
   l) Request record is not found (For cancelRequest)
   m) Insufficient command arguments (For all commands, eg. missing member id to register)
   n) Unknown command (Checking in the main loop in `main`())

   - Most of the above should be done by Exception Handling. You should name all Exception classes with prefix: "Ex", eg. "`ExBookIdInUse`", "`ExMemberNotFound`"

6. Refer to page 3 and the test cases for the requirements in each phase and the exact outputs.

   You may copy given code for text contents from the following file (also available on courseweb):



Requirements of Each Phase:

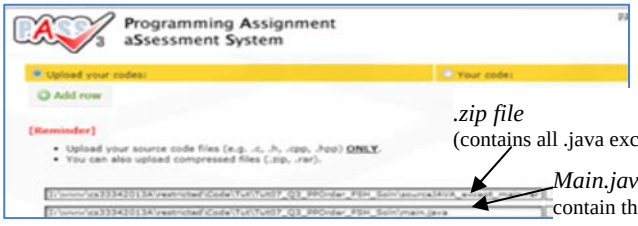
Also required: good programming styles + OO programming skills

Phases	Execution of commands	Requirements of proper undo/redo	Requirements of Exception handling / checking	Test case for reference (on courseweb)	Maximum Grade
Phase 1					
Phase 1.1	Register member, List members Start new day	--	Member ID already in use (For register)	1a.txt	~C-
Phase 1.2	Arrive new book List books Start new day	--	Book ID already in use (For arrival of new book)	1b.txt	~C+
Phase 1.3	1.1 + 1.2 + checkout (borrow book)	--	--	1c.txt	~B
Phase 1.4	Same as above	undo/redo of Register member Arrive new book Checkout	--	1d.txt	~B
Phase 1.5	Same as above	--	Member ID already in use (For register) Book ID already in use (For arrival of new book) Member not found (For checkout) Book not found (For checkout) Book borrowed by others (For checkout) Quota checking (Each can borrow 6 books only)	1e.txt	~B+
Phase 1.6	Assorted (1.1-1.5)			1f.txt	~B+
Phase 2					
Phase 2.1	Phase 1 items + checkin	undo/redo of checkin	Book not borrowed by this member	2a.txt	~B+
Phase 2.2	Phase 1 items + request	--	--	2b.txt	~A-
Phase 2.3	Phase 1 items + request	undo/redo of request	checking related to request	2c.txt	~A-
Phase 2.4	Phase 1 items + request + checkin + checkout	--	--	2d.txt	~A-
Phase 2.5	Phase 2.4+ + cancelRequest	undo/redo of cancelRequest	checking related to cancelRequest	2e.txt	~A
Phase 3					
Phase 3.1	StartNewDay: check onhold books which are due	(Assume no undo/redo for StartNewDay)	--	3a.txt	~A
Phase 3.2	request + checkin	undo/redo of checkin !!	--	3b.txt	~A+
Phase 3.3	Assorted			3c.txt	~A+

checkin : let the first queuing member to pick up in 3 days
checkout: pick-up the onhold book

Submission:

Please submit them to PASS as shown below:



.zip file
(contains all .java except *Main.java*)

Main.java (This class should contain the *main* method)

-- end --