Final Project

Due Mar 15 by 11:59pm **Points** 100

This week's makefile is here. Don't forget to rename it.

You are not required to submit anything written for the final project - just your finished code on TEACH. However, I hope you've all seen by now that making testing plans and designs before you start coding is very useful.

You will be writing a Library simulator. It will have three classes: Book, Patron and Library. To make things a little simpler for you, I am supplying you with the three .hpp files. You will write the three implementation files and another file that contains the main method, which will run a menu. You may add one-line "inline function" implementations to the .hpp files if you want, but implementation of the other functions should go in the corresponding .cpp files. Please put your function comments in the .cpp files (they would clutter up the .hpp files).

There are a few new things you will need to know about:

1. The vector::erase function lets you delete an element of a vector, shifting over all the elements after it.
2. If the holdings or members vectors get resized, that can invalidate any pointers to the Books or Patrons contained therein. Please use the vector::reserve function in the Library constructor to increase the capacity of the holdings and members vectors to 100.
3. NULL is a special value that signifies that a pointer does not have an actual value.
4. Near the top of the header files, you will see the word "class" followed by the name of a class that the header file needs to know exists, e.g. "class Patron;". This is called a "forward declaration", and it reassures the compiler that you intend to give it the actual definition of that class at some point.

Here are the .hpp files: Book.hpp, Patron.hpp and Library.hpp

Here are descriptions of the three classes:
Book:

- idCode - a unique identifier for a Book
- title - cannot be assumed to be unique
- author - the title and author don't need set methods, since they will never change after the object has been created, therefore these fields can be initialized within the constructor
- location - a Book can be either on the shelf, on the hold shelf, or checked out
- checkedOutBy - pointer to the Patron who has it checked out (if any)
- requestedBy - pointer to the Patron who has requested it (if any); a Book can only be requested by one Patron at a time
- dateCheckedOut - when a book is checked out, this will be set to the currentDate of the Library
- CHECK_OUT_LENGTH - constant that gives how long a Book can be checked out for
- two constructors - one for an empty Book, one that takes an idCode, title and author; checkedOutBy and requestedBy should be initialized to NULL; a new Book should be on the shelf
- some get and set methods (you may need to add others)

Patron:

- idNum - a unique identifier for a Patron
- name - cannot be assumed to be unique
- checkedOutBooks - a list of Books that Patron currently has checkedOut
- fineAmount - how much the Patron owes the Library in late fines
- two constructors - one for an empty Patron, one that takes an idNum and name
- some get and set methods (you may need to add others)
- addBook - adds the specified Book to checkedOutBooks
- removeBook - removes the specified Book from checkedOutBooks
- amendFine - a positive argument increases the fineAmount, a negative one decreases it

Library:

- holdings - a list of Books the Library has
- members - a list of Patrons the Library has
- currentDate - stores the current date represented as an integer number of "days" since the Library object was created
- a constructor that initializes the currentDate to zero
- addBook - prompts the user for book info (including the ID, which can be any arbitrary string) and uses it to create a Book, which is added to holdings
- addMember - prompts the user for patron info (including the ID, which can be any arbitrary string) and uses it to create a Patron, which is added to members
- checkOutBook
  - if the specified Book or Patron are not in the Library (an ID doesn't match an object in the Library), print out that message and return to the menu
  - if a Book is already checked out, print out that message and return to the menu
  - if a Book is on hold by another Patron, print out that message and return to the menu
  - otherwise update the Book's checkedOutBy, dateCheckedOut and Location; if the Book was on hold for this Patron, update requestedBy; update the Patron's list; print out that *Book title* has been checked out to *Patron name*
- returnBook
  - if the specified Book is not in the Library, print out that message and return to the menu
  - if the Book is not checked out, print out that message and return to the menu
  - update the Patron's list; update the Book's location depending on whether another Patron has requested it; update the Book's checkedOutBy; print out that *Book title* has been returned
- requestBook
  - if the specified Book or Patron are not in the Library, print out that message and return to the menu
  - if the Book is already requested by another Patron, print out that message and return to the menu
  - If the Book is currently checked out by the same Patron who is trying to request it, print out that message and return to the menu
  - update the Book's requestedBy; if the Book is on the shelf, update its location to on hold; print that *Book title* is on request for *Patron name*
- payFine
  - if the specified Patron is not in the Library, print out that message and return to the menu
  - use amendFine to update the Patron's fine; print out that the fines for *Patron name* are now *Patron fineAmount*
- incrementCurrentDate
  - increment current date; increase each Patron's fines by 10 cents for each overdue Book they have checked out (using amendFine)
- viewPatronInfo
  - if the specified Patron is not in the Library, print out that message and return to the menu
  - print the Patron's ID, name, the titles of any checked out Books, and their current fines
- viewBookInfo
  - if the specified Book is not in the Library, print out that message and return to the menu
  - print the Book's ID, title, author and location; the name of the Patron whom it's requested by (if any); the name of the Patron whom it's checked out by (if any), and the number of days left till it's due (or the word "overdue").
- **be careful - a Book can be on request without its location being the hold shelf** (if another Patron has it checked out);

Menu:
Give the user an option for each of the Library functions (except the constructors) and the option to quit.

Input validation: You should guarantee that book IDs and patron IDs are unique. If an object of that type already has the given ID, print "That ID is already in use." and return to the menu.

You must submit on TEACH: the provided makefile, the provided .hpp files, **Book.cpp**, **Patron.cpp**, **Library.cpp** and **Menu.cpp**.

To think about: There are six possible changes in the location of a Book. Can all six occur?

There are a lot of details here - it's entirely possible I omitted something or mis-stated something. If you think that is the case, I encourage you to ask me about it.

◂ Previous