ESE224 Fall-2017 Project

Library Management System

Table of Contents

1. Introduction

About the Project:

The final project of this course this semester is the library management system. It is supposed to execute common functions of the school library in reality. A user should be able to use his own username and password to access the system and perform corresponding operations according to his user type. For example, a reader can borrow a book and a librarian can add a new reader.

Project Objective:

- Design a project using C++ programming language;
- Implement object-oriented programming concepts such as inheritance, overloading, classes, objects, methods;
- Manipulate different types of data using appropriate data structure such as list, queue and vector;
- Perform input / output of data from / to files;
- Try to apply advanced concepts such as template;
- Practice analyzing and debugging techniques;
- Develop good coding habits:
    - Use separate code files for different classes' declaration and implementation;
    - Use functions to make the code concise and modular;
    - Use comments where necessary;
    - Use appropriate indent during coding;

Team Configuration:

You can form a group of 3-4 students to work on this project together. Each group is required to submit a soft copy of your source code (.cpp, .h and corresponding data files) and a project report. Detailed requirements will be given later in this file.

2. System Architecture

2.1 Infrastructure

As a reference, here is the infrastructure of the library management system including some necessary classes, you need to at least declare and implement them in your project. The details of each class will be talked about later in this chapter.

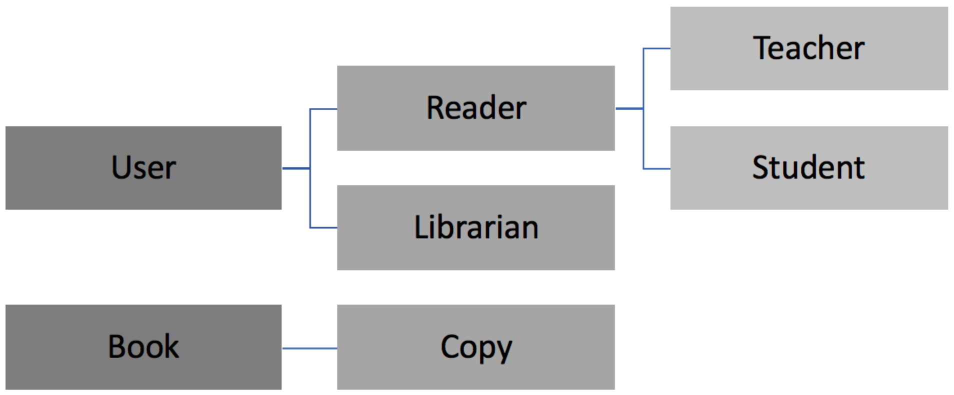


Fig 2.1 System Infrastructure

2.2 Class: User

User is a base class used to represent all the users of the system. Either a reader or a librarian can be called as "user". So, ***User*** class should have those common member variables and functions of all kinds of users. Specifically, user's **username** and **password**, as well as their accessors and mutators, should be included.

2.2.1 Class: Reader

Reader is a derived class of ***User*** class. Each object of this class represents a reader of the system. In reality, a reader can perform operations such as borrowing a book, returning a book and reserving a book, so ***Reader*** class should keep information such as:

- Reader's Username;
- Reader's Password;

- Maximum number of copies that a reader is allowed to keep at the same time, it'll be reduced because of penalties;
- List of copies borrowed by the reader;
- List of books reserved by the reader;
- Number of penalty, overdue copies will lead to penalty;

Apart from accessors and mutators, some other member functions are needed to execute these operations. These will be introduced in the next chapter.

Also, ***Reader*** class can be divided into two types: ***Student*** and ***Teacher***.

2.2.1.1 Class: Teacher

Teacher class should have all members defined in ***Reader*** class, also

- A teacher can keep more copies simultaneously, e.g. 10.
- A teacher can keep a copy for a longer period, e.g. 50.

2.2.1.2 Class: Student

Student class should also have all members defined in ***Reader*** class, and

- A student can keep less copies simultaneously, e.g. 5.
- A student can keep a copy for a shorter period, e.g. 30;

2.2.2 Class: Librarian

Librarian is the other derived class of ***User*** class. Each object of this class represents a librarian of the library. The information needed to be kept in this class is the same as in ***User*** class. However, a librarian can perform more operations such as adding and deleting books or users. These will be talked about in detail in the next chapter.

2.3 Class: Book

Book class is used to represent a set of the same copies of a specific book. A book has its own unique ISBN, and although there're many copies having the same ISBN, you should only create one object of ***Book*** class. Each object of this class should have the following information:

- ISBN, a sequence of digits to uniquely identify a book;
- Book's title;
- Book's author. For simplicity, each book has only one author;
- Book's category, such as math, chemistry, engineering;
- List of reservers, indicating who reserves the book and if any copy of the book is available, the reserver can borrow it. <u>You can implement it using a queue;</u>

2.4 Class: Copy

Copy class is a derived class of ***Book*** class. Since there may be several copies of the same book, each copy can be represented by an object of this class. Therefore, ***Copy*** class should include all the member variables of ***Book*** class. Besides, the following information should be kept:

- ID. This is the unique identification for each copy;
- Reader's name, indicating who borrows this copy;
- borrow date, indicating when the reader starts to keep the copy;
- reservation date, indicating when the reader starts to make the reservation;
- expiration date, indicating the expected date for the reader to return the copy;

3. System Functions

In this chapter, you're supposed to implement the following functions. The ones followed by "[optional]" belong to bonus part.

3.1 File Operation

In your project, all the data should be stored in files, specifically, reader.dat, librarian.dat, book.dat and copy.dat. Each time you run the program, it firstly reads these files and all the data should be loaded into appropriate data structure, such as vectors and lists. And when the program is ended, the data should be updated back to the same files.

Note:

- You're required to implement overloading of the stream operator "<<" and ">>" for each class and use them in this function. Otherwise, you can only get half scores of this function;
- You're recommended to use function template here, because given the filename, the loading and saving are the same with the help of overloading. If you use it, you can get bonus. [optional]

3.2 User Authentication

The user of the system has his own username and password. Each time he wants to access the system, he is asked to enter them for authentication. Only if it's successful, he can access the system and perform operations. Otherwise, the program will be directly ended. According to the user's type, your system should provide different menus.

Note:

- **Password Mask:** When users enter their passwords, there should be "*" shown on the screen instead of actual character of the passwords. [optional]

3.3 Date Counter

In reality, each copy can be kept for a certain period of time, e.g. 30 days. So, in your system, you should set up a counter to simulate the calendar. You may call the function **clock()** of **<ctime>** library to get the clock ticks. The usage of this function and a simple example are given on http://www.cplusplus.com/reference/ctime/clock/. And in your system, you need call this function to get the start time at the beginning of your program:

clock_t start = clock();

Also, you have an initial value of your counter, e.g. *ini_date = 30*. Then, each time you want to get the current date, you can call the function again to calculate it:

clock_t cur = clock();

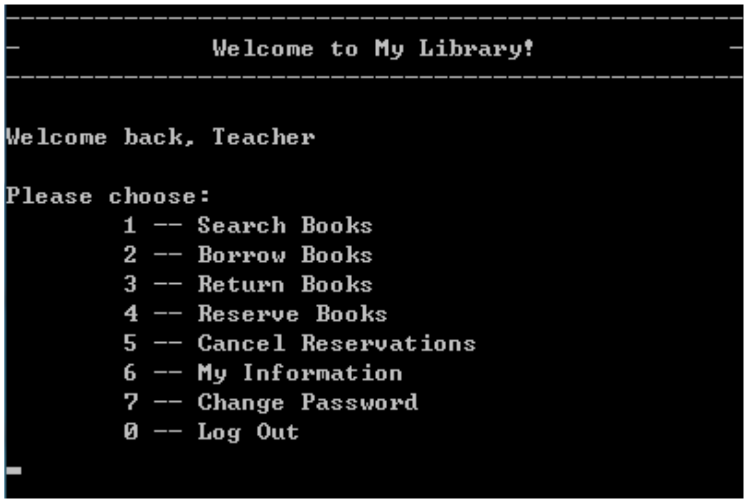
cur_date = ini_date + int(cur - start) / CLOCKS_PER_SEC;

Here, *CLOCKS_PER_SEC* represents the number of clock ticks per second and you can directly use it. In the given case above, the counter increase by 1 every second, i.e., a “day” is 1 second.

Note:

- In your system, the length of a “day” is about 5 seconds;
- Each time you access the system, the date should be counted continuously from the ending date of last access.

3.4 Reader’s Operations



```
-------------------------------------------------
-               Welcome to My Library!          -
-------------------------------------------------

Welcome back, Teacher

Please choose:
        1 -- Search Books
        2 -- Borrow Books
        3 -- Return Books
        4 -- Reserve Books
        5 -- Cancel Reservations
        6 -- My Information
        7 -- Change Password
        0 -- Log Out

_
```

Fig 3.1 Reader’s Menu

Note: The “book” here means a copy of books except 3.4.1 and 3.4.7.

3.4.1 Search Books

The user is allowed to search books by keys, including: ISBN, title, author's name and category. The output should include all the basic information of the searching result such as ISBN, book's title, author's name and category. Besides, The IDs of available copies of the book are also needed to be listed.

Note:

- If several books share the same author or belong to the same category, the system should display them all. In this case, the searching results should be sorted by their popularity. The popularity is measured by the length of the reserver list of the book.

3.4.2 Borrow Books

In your system, the reader is able to borrow a copy of books by identifying the ID of the copy. In this case, this reader should be assigned as the current reader of this copy. Also, the copy should be added to the reader's list of borrowed copies.

Note:

- If the reader has overdue copies, he cannot borrow new copies;
- A reader cannot borrow more copies than the maximum allowable number;
- A reader cannot borrow a copy which is reserved by others. Only if the reserver list of the books is empty, or this reader is the first one of the list, he can borrow it;
- If several copies of a book is available, the first reserver can borrow any of them;
- A reader cannot borrow a copy that has been lent to others;
- The length of the period that the reader can keep a copy for is determined by the popularity of the book. For example, if no one reserve the book, a student can keep a copy for 30 days. But if 30 readers reserve the book, the length should be shorter, such as 20 days. So, you need calculate the expiration each time the reader borrows a copy. [optional]

3.4.3 Return Books

The reader can return the copy he borrowed. If successful, the copy becomes available again and it can be borrowed by the first one in reserver list. Also, the copy should be removed from this reader's list of borrowed copies.

Note:

- Although the reader returns the overdue copies, the number of his penalty should increase by one. If the number of penalty exceeds the threshold, the maximum number of copies he can borrow needs to be reduced.

3.4.4 Reserve Books

In your system, the reader needs to be able to make a reservation for a book if all the copies of it are lent out to others. And then, this reader is added to this book's reserver list.

Note:

- If a reader has overdue copies, he cannot reserve any books;
- If the reserver list is not empty, the first one is always the valid one to borrow the book's copies;
- If there is any copy available, but the first reserver doesn't borrow the book. Then his reservation will be cancelled automatically after 10 days.

3.4.5 Cancel Reservations

Correspondingly, the reader can cancel his reservation of a book. Then, he is removed from the book's reserver list. Also, the book should be removed from the reader's reservation list.

3.4.6 Renew Books [optional]

Renewal of the copy will only occur in the case that the reader has borrowed a copy and he wants to keep this copy for longer. Then before the expiration date, he can choose to renew it.

Note:

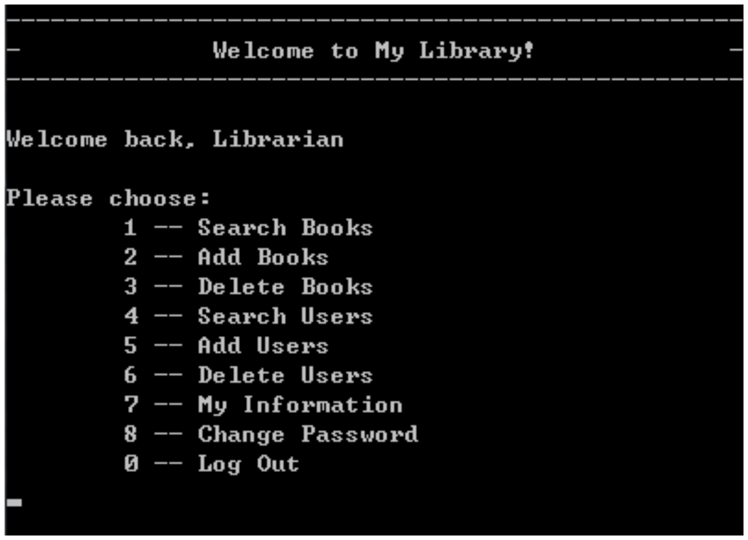
- Only if the book's reserver list is empty or the reader is the first one in the list, he can renew this book's copy;
- A reader cannot execute the renewal operation if he has copies overdue, including the one he wants to renew.

3.4.7 Recommend Books [optional]

Your system should be able to recommend books to readers if they want. If the reader's list of borrowed copies is not empty, then according to the category of the last copy he borrowed, you should recommend the 10 most popular books with the same category. If the list is empty, the 10 most popular books should be recommended regardless categoty.

Besides, a reader can also choose to show his own information, including username, password, list of copies that he is keeping, etc., and change his password. They're easy to be implemented by class accessors and mutators.

3.5 Librarian's Operations



```
-----------------------------------------------------------
-                  Welcome to My Library!                 -
-----------------------------------------------------------

Welcome back, Librarian

Please choose:
        1 -- Search Books
        2 -- Add Books
        3 -- Delete Books
        4 -- Search Users
        5 -- Add Users
        6 -- Delete Users
        7 -- My Information
        8 -- Change Password
        0 -- Log Out

_
```

Fig 3.2 Librarian's Menu

Note: The "book" here also means a copy of books, except 3.5.1.

3.5.1 Search Books

It's the same as the one for readers.

3.5.2 Add New Books

The librarian can add new copies to the library. He need to give all the information of the new copy, including ISBN, book's title, author's name, category. And then the system should assign a new ID to this new copy.

Note:

- If there aren't other copies of the same book as the added one, you should also create a new object of this book.

3.5.3 Delete Old Books

The librarian can also delete the old copy from the library by giving the ID of it.

Note:

- If the copy is lent out, it cannot be deleted;
- If the copy is deleted, it should be also removed from the readers' reservation list;
- If the target is the last one copy of the book, you should also remove the book.

3.5.4 Search Users

Similar to searching books, the librarian can search users by their usernames. All the information of the searching results should be displayed.

- If the target is a librarian, show his username and password only;
- If the target is a reader, show his name, password, reader type as well as the copies he is keeping currently;

3.5.5 Add New Users

The librarian has the privilege to create accounts for new users. After choosing the type of the new user, he only need to provide the username and password of this user.

Note:

- The name of the new user should be different from those of existing users;

3.5.6 Delete Old Users

Also, the librarian can delete old users from the system by providing their names.

Note:

- If the reader keeps any book copies, he can't be deleted;
- After deletion, the reader needs to be removed from the corresponding books' reserver list;

Also, a librarian can also choose to show his own information and change his password. They're easy to be implemented by class accessors and mutators.

4. Requirements & Grading

4.1 Submission Requirements

4.1.1 Code Requirements

- All codes should be completed by yourselves. TAs will read your codes, and copying from other groups or other resources is definitely forbidden;
- Make sure your codes can be compiled successfully before you submit them;
- Your codes should be concise and modular. All files, including .cpp, .h and all data files, should be submitted in a project folder, and all codes in one main file is not recommended;
- Provide comments in your codes where necessary;
- In your system, there should be at least one librarian account so that TAs can test your codes;
  - Username: TA224
  - Password: 224

4.1.2 Report Requirement

Please pay attention to your report. TAs will first read your report first and check your codes according to it. Your project report should describe clearly:

- Architecture of your project;
- Specification of classes, including the member variables and functions of each class;
- System functions you implement, including how parameters are passed when each function is called;
- Usage of advanced concepts, such as template, overloading;
- All the highlights of your projects;

4.2 Grading Criteria

The whole project is divided into three parts:

- Required part (80 points)
  - Implementation of all classes (10 points)
  - File operation (5 points)
    - Usage of overloading (5 points)
  - User authentication (5 points)

    - Date counter (5 points)
    - Reader’s operations (25 points)
    - Librarian’s operations (25 points)
- Bonus part (30 points)
    - Concise and modular coding style (5 points)
        - Putting different classes’ declaration and implementation in separate files
        - Giving comments in your codes
        - Using functions instead of putting all lines in main() function
    - Usage of template (5 points)
    - Extra functions (5 points / each)
        - Recommend books
        - Renew books
        - Adjust expiration date according to book’s popularity
        - Password mask
- Report (20 points)