Li

Library Management System

A C++ project made for CST2550

Maciej K Ciba M00864763

Middlesex University LOndon

1. The project

The main goal of the project was creating a simple, working management system for a library. In the given scenario the librarian would use the software to:

* Add new member and display their details directly after
* Issue a book with a valid 3 days due date
* Return a book from a member
* Display all books currently borrowed by certain user
* Calculate a fine for overdue books upon returning them

The project has to simplify librarian flow of work by allowing them to use the features implemented to see what books can be rented to users, what’s the fine for overdue books etc.

1. The implementation

After understanding the requirements and marking sheet I immediately understood that I will be forced to use more methods than there are in original UML. I came up with two new helper classes to help me take care of dates and some global methods. After creating the UML I started implementing the classes, starting with their header files that are stored in header folder and then .cpp files stored in src folder.



1. Main functionalities
2. Adding Members: after logging in the Librarian can add new Members, they have to select:

- a valid string name,

- an address

- a valid email address

If their inputs are not valid the helpful message is displayed telling the user what

need to correct. After inputting all the necessary data in, the new Member details are displayed.

1. Issuing a Book: librarian can issue a book if they have valid members. They must select their ID and ID of the book the member would like to loan. After inputting the correct ID’s, the messages are displayed saying what book was rented to what member, what is the current date and what is the due date.
2. Return a book and calculate the fine: if a member has loaned a book, they can return it. Upon returning it the software will take the due date from book records and compare it with todays’ date if a todays’ date is larger is calculates the fine £1 a day.
3. Display Borrowed books: if a member has any books loaned, the librarian can display all of them after entering members’ ID.
4. Testing

I used two methods for testing, I tested manually and automatically. My manual tests were conducted on every part of the code ( especially when it came to user inputs). For more code testing I used catch2 provided by university to test my software. My automatic testing included testing:

* the initialization of my classes works properly
* the books are properly borrowed and returned
* the current date is a current
* the dates between method calculates the correct number of days

Here are my results of my test cases:

A screen shot of a computer

Description automatically generated

1. Usage of Github and version control

Throughout the process of implementing the code I used github for my version control. I found it very helpful when reverting changes that messed up most of the code. I created alias coa so I don’t have to add all files in the folder one by one, but I can add all of them and commit with a single command line. I “pushed” every major change that happened, it accumulated to over 14 commits over the span of 2 weeks.

A screenshot of a computer

Description automatically generatedA computer screen shot of a program

Description automatically generated

1. Makefile

I used makefile to compile all my C++ files. It defines rules for building an executable from source code, organizing object files (put all of the in a bin folder that it creates), and cleaning up the build environment (with make clean).

1. Conclusion

Summary:

-the project works, it allows librarian to use all the functions that were required. In my personal opinion I think that code is well organized with adequate number of comments and spacings.

- the project was pushed to GitHub very often to minimize the possibility of the entire project crashing.

-catch2 tests are working as expected, no failed tests.

Limitations

* More OPP could’ve been used more.
* Optimization could’ve been better.

Notes for the future:

* Start working on a project sooner
* Give myself more time to think thoroughly think about each method and what role are they going to play
* Commit a bit more (initially I wanted to make it to 20 commits)