Library System

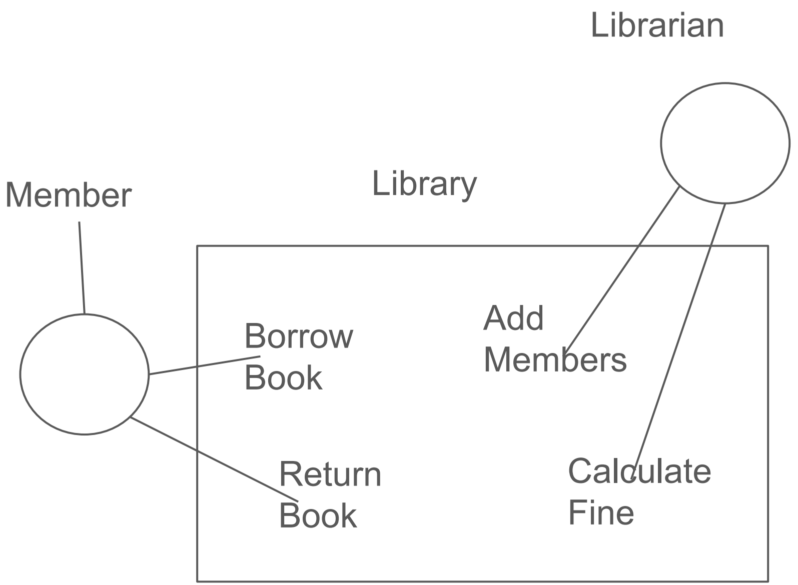
Name: Carlos Bacchus

Student Number: M00862705

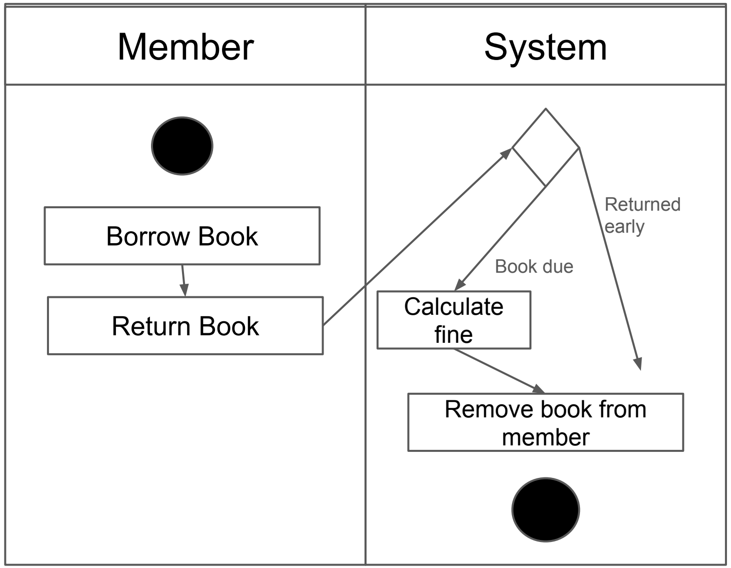
I have been tasked with designing and implementing a library system that will enable a Librarian to manage the loaning of books.

Design

Use case



Activity



Implementation

I started by creating the person parent class as header and cpp files. Then I made the Library and member classes and had them inherent from the person class.

When I created the Book class I needed to use forward declaration to ensure there weren't errors

Then I started implementing my main cpp file called Library:

I created an enum system for each page in the system

I then created functions for searching the file containing book data and creating instances of the Book class.

Then I used a makefile to compile Library.cpp and tested it using the console.

Testing

I tested the code through frequent output of variables to console

Issues:

Circular dependency- Book and Member classes had a circular dependency

Solution: use forward declaration

Multiple Definitions of the person class

Solution: put the shared definitions in a singular file

File not reading correctly with the book type at the end of line

Issue with removing a book from the members list

Solution: I had to make a new removal function for the member class that wasn’t outlined in class diagram

Some of the functions required certains tests such as:

whether there were any members available

or

whether the members had any books

Demonstration

The software starts by having the user input their information and creating a Librarian class with it

A page enum will switch to the home screen displaying all their options

The user will then be taken to the home screen with 6 options

Add member, issue book, return book, display book, quit and pass the day

If there are no members the user can only add members, pass the time or quit

Pass the time increases the time of day to simulate time passing

Issue book takes user input to search through the book file by type and then by name. It will add a new book to a member’s vector

Return book checks whether the book is late or not by comparing the due date to the current day then issuing a fines based on that difference if the due date is lower than the current day

Display outputs all the books a member has using a for loop and vector

Add Member creates a new member using user input and puts it in a vector called “members”

Conclusion

In conclusion I believe I have Implemented the necessary features for this program. Though in the future I will spend more time on the early planning phase and testing rather than attempting to implement it without necessary design and preparation as I had to make numerous alterations to the classes to fit with the implementation.