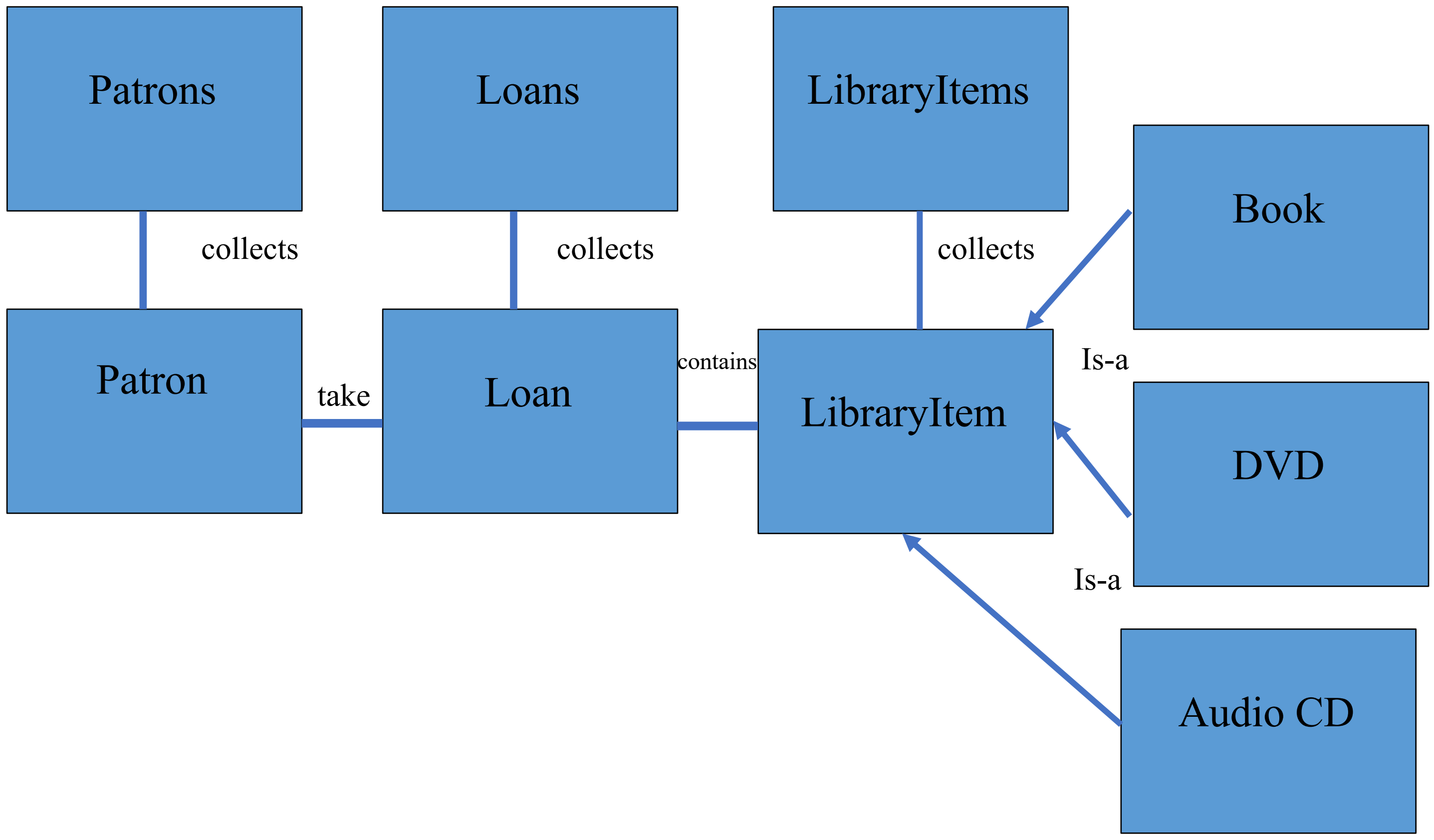
Suzanne Nolba

Homework #4

CSCE 1040 – Computer Science
Section 001

Class Relationships



Class Contents

Patron

Name (string)
idNumber (int)
fineBalance(float)
Curr_Num_Books_Out(int)
Set/Get Name()
Set/GetIdNumber()
Set/GetFineBalance()
Set/GetCurr_Num_Books_
Out()

Loan

Loan_Id(int)
LibraryItem_ID(int)
patronID(int)
Due_Date_Time(int)
Loan_Curr_stat(string)
Set/GetLoan_ID()
Set/GetLibraryItem_ID()
Set/GetLoan_curr_stat()

LibraryItem

Author(string)
Cost(float)
Status(string)
Loan_period (int)
Set/Get Author()
Set/Get Cost()
Set/Get status()
Set/Get Loan_period()
Ostream& operator<<()
Virtual prinInfo()
Virtual rawPrint

Book

Title(string)
Author(string)
ISBN Number(int)
Category(string)
Set/Get Title()
Set/Get Author()
Set/Get ISBN_Number()
Set/Get Category()
printInfo()

Audio CD

Artist(string)
Title(string)
Number_tracks(int)
ReleaseDate(string)
Genre(string)
Set/Get Artist()
Set/Get Title()
Set/Get Number_tracks()
Set/Get ReleaseDate()
Set/Get Genre()
printInfo()

DVD

Title(string)
Category(string)
Run time(int)
ReleaseDate(string)
Studio(string)
Set/Get Category()
Set/Get Title()
Set/Get RunTime()
Set/Get ReleaseDate()
Set/Get Studio()
printInfo()

Patrons

PatronList(vector)
Add_patron()
Edit_patron()
Delete_patron()
Payfine()
Search_patron()
PrintPatrons()
PrintPatron()
EditNum_books_out()
Edit_fine()
loadPatron()
storePatron()
cleanup()

Loans:

LoanList(vector)
Check_Out_Item()
Check_In_Item()
Overdue_list()
Item_Patron_List()
Update_Loan_stat()
Re_check_Book()
Edit_Loan()
ReportLost()
storeLoan()
loadLoan()
cleanup()

LibraryItems

ItemList(vector)
ItemCount()
Add_Item()
Edit_Item()
Edit_ItemStat
Delete_Item()
Search_Item()
PrintItems()
PrintItem()
loadItem()
storeItem()
search_item_availability()
item_cost()
access_loanPeriod()

Function Pseudo Code

```
Add Patron:

    -   Prompt user for name
    -   Prompt user for ID Number
    -   Prompt user for fine balance
    -   Prompt user for Current number of books out
    -   Create Patron object
    -   Populate Patron’s object
    -   Add object to collection

Edit Patron:
    -   Prompt user for ID number
    -   Create Patron object
    -   Check if patron ID number match any in the collection
    -   If it does, prompt user for data he/she would like to replace (name…)
    -   Replace the old data by the new one
    -   Save object modifications to the collection
Edit_fine
    -   Check if the patron ID number that was passed match any in the collection
    -   If it does, add the new fine amount to the fine balance -   exit
Edit_NumBooksOut
    -   Search if the patron Id match any on the patron collection
    -   Increment (if loan was successful) or decrement(when checking in) the
        current number of books out of the patron at that position Delete Patron:
    -   Prompt user for ID Number
    -   Create a Patron object
    -   Search if ID number enter by user match any ID number in the collection,
            •  Delete the object from the collection


PayFine:
    -   Create a patron object
    -   Prompt user for patron ID
    -   Search if patron ID match any ID in the collection
    -   Display the current fine balance
    -   Prompt patron the amount he would like to pay
    -   Proceed the payment
    -   Update the current fine balance
    -   Save the modification in the collection
```

Search Patron:

- Prompt user for ID number
- Create Patron object
- Search if ID number match any ID Number in the collection
    - Return the index of patron at which the patron is in the collection

Print Patrons:

- Create a Patron object
- For each patron in the collection, read object name, ID Number, Fine balance, Current number of books out.

Print Patron:

- Create a Patron object
- Search for index of Patron in the collection using patron ID Number.
- Read object name, ID Number, Fine balance, Current number of books out for patron at the specific position in the collection

Add Library Item:

- Create item library ID
- Get item cost
- Get item status
- Get item loan period
- Determine what library item it is
- Create item object
- Populate item object
- Add object to catalog
-

Delete Library Item:

- Prompt user for item library number
- Create item object
- Search if ID match any in the catalog
- Delete the object from the catalog

Edit Library Item:

- Prompt user for item ID number
- Create item object
- Check if the id number match any in the library item collection
- Replace the old data by the new one
- Save the modification in the catalog

Search Library Item:

- Prompt user for item library Id
- Create item object

- If id number match any in the catalog, return index at which the item was found

Search item availability:

- Search if the item id that was passed match any in the catalog
- Check the item status
- Return false if current status is Lost or Out
- Return true if current status is In

Print Library Items:

- Create library item
- For each item in the library item collection, read object data

Print Library Item:

- Prompt user for item id number
- Create item object
- Read object data at the specific position.

Check Out an item:

- Pass an object patron
- Pass an object LibraryItems
- Create a Loan object
- Prompt user Patron ID
- Check if patron ID is found in the list of overdue
    1- If it is found, stop the check-out
    2- If not, proceed the check-out
- Check patron's number of books out
- If number of books out <=5, proceed the check out
- If not, stop the check out
- Create the loan ID
- Set loan's current status
- Determine the type of item patron wants to loan
- Populate the Loan object
- Add the object to the Loan collection

Check in an item:

- Prompt user for the Loan ID
- Create Loan object
- Pass LibraryItems object
- Pass patrons object
- Look if the Loan ID match any in the collection
- Check Item's condition and update item's current status
- Update patron's fine balance
- Delete object from the Loan collection

List of all the overdue:

- Create a loan object

- Search for the object loan's current status
- Store object to the list if status is overdue

List of all items for a particular patron:

- Create a loan object
- Pass a patron object
- Prompt user for the patron's ID number
- Look if patron ID match any in the loan collection
- Read all the items loaned by the patron
- Store in the collection

Update loan status:

- Create a loan object
- For every second: look inside loan collection
- Get the present time in second
- Get the due date and convert it in second
- Compute due time minus present time:
  1- If due date (in time_t) > today's date (in time_t) set the loan status to normal
  2- If due date (in time_t) < today's date (in time_t), set loan status to overdue.

Re-check an item:

- Prompt user for the Loan ID
- Create Loan object
- Check if Loan ID number match any in the collection
- If it does, add 10 to the loan due date (for book item only)
- Update the old due date by the new one
- Save object modification to the collection

Edit a loan:

- Prompt user for the loan ID
- Create Loan object
- Check if Loan ID number match any number in the collection
- If it does, prompt user for the data he/she would like to edit
- Replace the old data by the new one
- Save object modification to the collections
-

Report a loan:

- Create a Loan object
- Create a LibraryItem object
- Create a patron object
- Prompt user for the loan ID
- Look if loan ID match any in the collection
- Update item's current status
- Update patron's fine balance
- Save the object modifications in the collection

Design Project Report:

For this project, I was asked to design and create a system to manage loans from a public library. The program in question verifies certain criteria in order to proceed to a loan. The local public library offers loans for Audio CD, book, and DVD. In order for a patron to be eligible for a loan, he has to have less than 6 books out at the moment of the new loan, and also not have any overdue of any item in the library. The system was design in a way that the library will only offer the recheck option for book not any other item. Only one recheck with an extension of ten days is possible. The loan periods are: 2 days for an audio CD, 3 days for a DVD, and 10 days for a book.

The program will also manage the fines' payments (fines someone is necessary, update all fine payments…). To make this feasible, I created 9 classes. Patron, Patrons (that holds the collection of all patrons), LibraryItem (based class for all items that will be find in the library), LibraryItems (to hold all the library items), Book, AudioCD, DVD, Loan, Loans (holds collection of all loans).

I have decided to add several functions to ease the use data from one class to another. I added a function that will either decrement or increment the number of books out per patron depending on either he/she is checking out or in. I also added function to check the availability of each item in the catalog.

While designing this system, I have assumed that only the status (especially if an item such as a book was in repair, and we would like to change the status from "repair" to "In") and cost of any item presents in the library can be modifiable. As for the patron, the system will allow the user to edit only the name of the patron because all the other information will be modified automatically.

Every time the program starts, a load function is called. It loads each collection with the data for each collection class. Allowing the user or system to keep track of everything that was done previously. At the end, a store function is called. Allowing data to be stored in a file.