Ani Megerdichian
Darshana Palekar
Zhenning Guo

CIS22B Group Project – Requirement Analysis

For the project we are required to develop a software package for a bookstore. The program has to support three modules, namely

- Cashier module – support all tasks associated with the sale of a book item.
- Inventory module – maintain an updated inventory of all book items.
- Report module – display all records related to the inventory.

Design Approach

Attributes of books are split into two classes; BookData and BookInfo. BookData holds the isbn, wholesale price, retail price, and quantity on hand. A class called Date is also created, which holds month, day, and year. BookInfo extends Date and BookData and holds the author, title, and publisher. An array of BookInfo objects is created to hold all of the books in the inventory.

Since the inventory is in a text file, we decided to read the file, line by line, using standard file I/O methods. Then we parsed each line, and applied the necessary conversions. Each line holds the information for a complete book record, and so after the conversions, the values are used to create a BookInfo object. An array of 25 BookInfo objects is created. The BookInfo array acts as a working inventory while the program is running.

It was decided to use a menu driven program using a switch statement that would allow the user to choose the appropriate module. For the modules, we decided to make each of the three modules its own class. A class called Module would act as the base module, and the Cashier, Inventory, and Reports classes would extend from it.Every module class has a function called module( ), which takes in a BookInfo array and an integer size as a parameter. This function is declared virtual in the Module class. Derived class objects are created of each module class, and Module class pointers are assigned to these objects.The module function of the appropriate module will be called during runtime when the user makes their choice, demonstrating polymorphism.

Each of the three derived modules is also menu driven, with a switch statement that allows the user to choose the appropriate actions for the specific module. These actions are controlled by functions that are defined within their respective classes. Search functions that all modules use are also defined in separate header files, including a template function for searching.

When the user decides to quit the program, a temporary text file will be created and the modified book objects will be written into this file using an overloaded insertion operator. The old inventory file is deleted, and the temporary file is renamed and acts as a new and updated inventory file.

The Cashier Module

This module allows the computer to act like a cash register. It is also required to support all operations associated with the sale of a book item.

Design Approach

The Cashier module must perform three tasks: keep track of the books the user would like to purchase, edit the quantities of these specific books, and display a report with calculations once the user decides to purchase.

A menu-driven program using a switch statement to allow the user to add, edit, and purchase a book was considered. The following functions were then written:

addToCart( ) - allows the user to add books for purchasing to a virtual "cart"

editCart( ) - edits the quantities of the books that were added to the "cart"

checkout ( ) - performs calculations, displays a report of transaction, and modifies the BookInfo array based on the changes made in the transaction

These functions utilize arrays to hold the information of the books that are considered for purchase. If editing is needed, these arrays are modified. A template function called findIndex( ), which returns the index of a value in an array, or returns -1 if not found is used to search these arrays. The BookInfo array is only modified once the user decides to complete the transaction. When user input is needed, these three functions include steps to validate user input and guide the user through the module. This allows for a better user experience, since the user is not brought back to the module menu after every incorrect input.

The Inventory Module

This module maintains an updated inventory list of all book items. The inventory list contains information pertaining to the following fields:

ISBN
Title
Author
Publisher
Date Added
Quantity on Hand
Wholesale Price
Retail Price

The module should update the quantities of the book items as they are sold/added. In addition, the inventory module should allow the user to look up information on any book in the file, add new books, delete books and update any information in the database.

Design Approach

An Object-oriented programming(OOP) approach that centers around the book object was considered. Information pertaining to the various fields were used as the book attributes and member functions that access the book object data were created. Since only the objects member functions may directly access and make changes to the objects data, the data remains hidden from outside code. This ensures data integrity and avoids accidental data corruption.

To achieve this goal, it was decided to first create an array of 25 book objects using a base class BookData. This class contains information related to ISBN, Wholesale Price, Quantity on Hand and Retail Price. A Date class was also created to store the Date Added field. The BookInfo class was then derived from the BookData and Date classes using the principles of multiple inheritance. The BookInfo class inherits from the BookData class and Date class via public access specifiers. It extends the BookData class and contains information on the title, author and publisher of the book. It was decided to use overloaded constructors to create and edit the book objects. Mutator functions were kept to a minimum. This limits the manipulation of the book object data and minimizes chances of data corruption and creation of orphan records. The inventory list containing the data on the books was stored in a text file named "Booklist.txt". The information from the text file was read and parsed into an array. The array members were passed to the overloaded BookInfo constructor and then used to populate the book objects.

A menu-driven program using a switch statement to allow the user to look up, add, edit and delete a book was considered. The following functions were then written:

LookUpBook() – to look up book info
AddBook() – to add a new book
EditBook() – to edit a book
DeleteBook() – to delete a book

The Report Module

This module is required to analyze the information in the Inventory Database to produce the following reports:

- Inventory List – a list of information on all books in the inventory.
- Inventory Wholesale Value – a list of the wholesale value of all books and the total wholesale value of the inventory.

- Inventory Retail Value –a list of the retail value of all books and the total retail value of the inventory.
- List by Quantity – a list of all books in the inventory sorted by quantity on hand, with the book with greatest quantity listed first.
- List by Cost – a list of all books in the inventory sorted by wholesale cost, with the item with greatest wholesale cost listed first.
- List by Age – a list of all books in the inventory sorted by purchase date, with the item with the greatest on shelf life listed first.

Design Approach

The inventory database module adopts an Object-oriented approach to create the book objects. These book objects then serve as the basis for the reports module. The attributes and member functions provide means for obtaining all relevant data relating to each book object. This information can then be used/ formatted to generate the appropriate reports.

A menu-driven program using a switch statement that allows the user to select the desired type of report was considered. The following functions were defined to handle each report request:

reportInventoryList() - lists all book objects and their attributes in inventory.

reportWholesaleValue() - lists the total wholesale cost per title (as product of cost per book and the quantity on hand), while an total accumulator maintains a running total of the inventory wholesale value.

reportRetailValue() - lists the total retail cost per title (as product of cost per book and the quantity on hand) and an total accumulator maintains a running total of the inventory retail value.

reportListByQuantity() - lists the title and units on hand for each book in descending order, with the greatest quantity listed first.

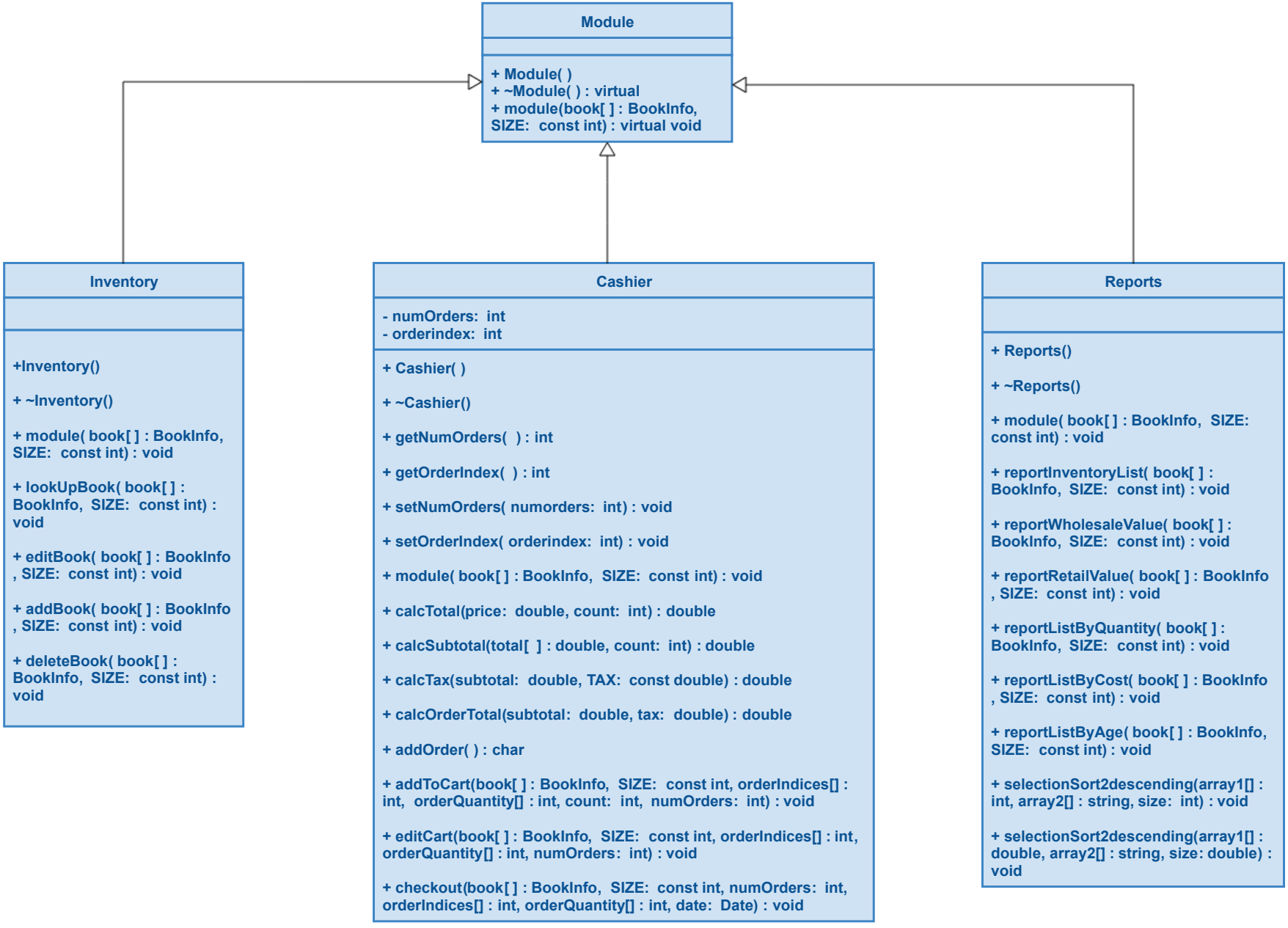
reportListByCost() - lists the title and wholesale cost for each title, sorted in a descending order. The items with the greatest wholesale cost is listed first.

reportListByAge() - lists the title and the age of the book, sorted in descending order based on purchase date.

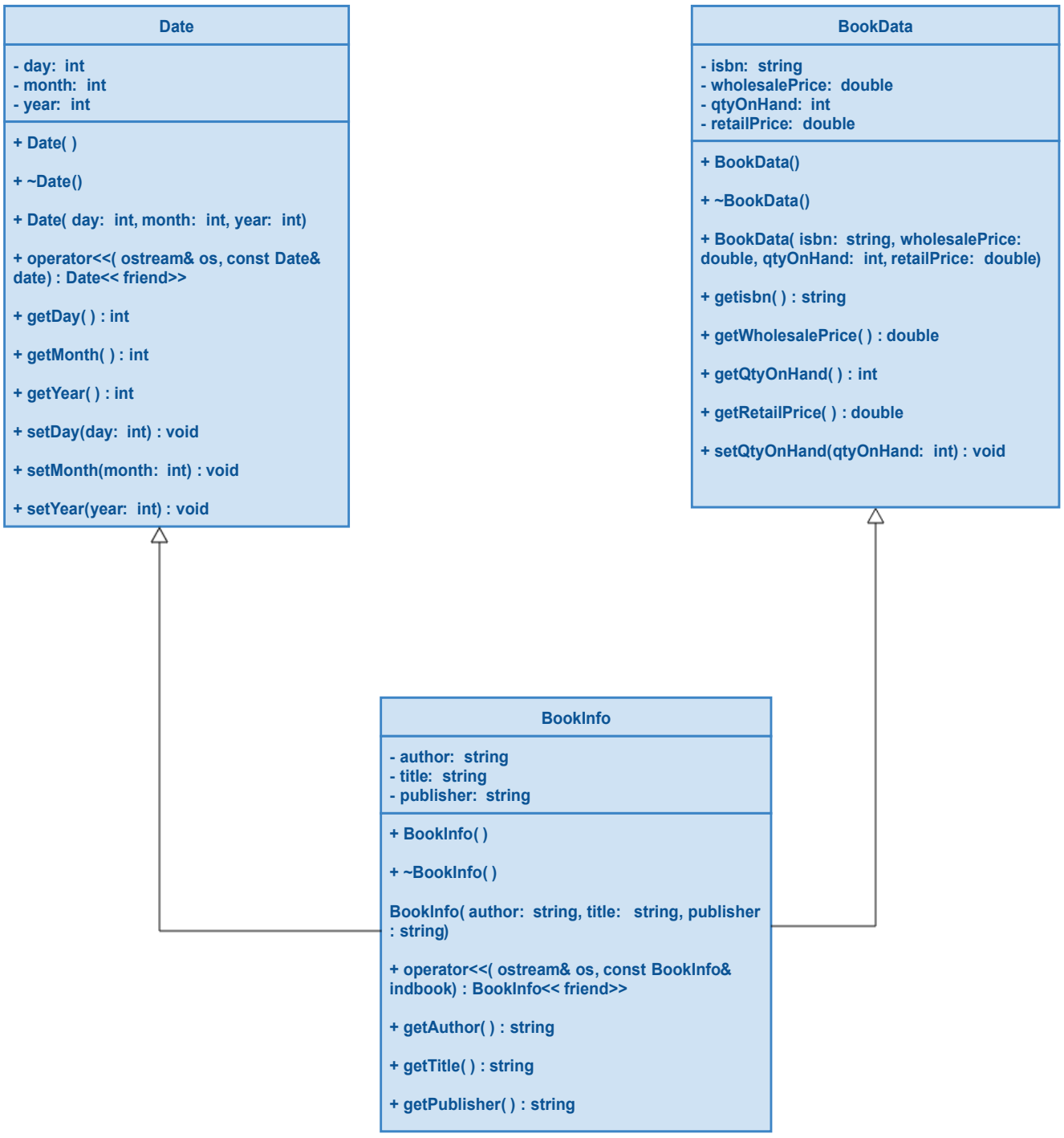
The book object array and size were passed as function arguments to the calling function. Accessor functions were used to obtain relevant attribute information and the data was stored in parallel arrays. The arrays were then sorted using the selection sort algorithm. The selection sort algorithm with same name but different parameter lists, demonstrates the use of overloaded

functions, where the compiler calls the appropriate functions based on the arguments of the calling function. The reports help the bookstore identify slow moving and fast moving inventory items. These reports serve as resourceful tools for inventory planning, budgeting and sales forecasting.

CIS22B Final Project: Serendipity Booksellers
UML Class Diagrams



CIS22B Final Project: Serendipity Booksellers
UML Class Diagrams



Z:\animegerdichian\Project\x64\Debug\Project.exe

```
Inventory accessed successfully.

                  Serendipity Booksellers
                          Main Menu
1. Cashier Module
2. Inventory Database Module
3. Report Module
4. Exit

Enter your choice: 2
                 Serendipity BookSellers
                   Inventory Database
 1. Look up a Book
 2. Add a Book
 3. Edit a Book
 4. Delete a Book
 5. Return to the Main Menu

 Enter your Choice : 1
You selected look up book.
Enter the ISBN of the book you would like to look up: 1119055806

ISBN: 1119055806
Title: R for Dummies
Author: Andriede de Vries
Publisher: John Wiley & Sons
Wholesale Price: 21.99
Quantity on Hand: 6
Retail Price: 29
Month added: 3
Day added: 1
Year added: 2017
                 Serendipity BookSellers
                   Inventory Database
 1. Look up a Book
 2. Add a Book
 3. Edit a Book
 4. Delete a Book
 5. Return to the Main Menu

 Enter your Choice : 2
You selected add book.
Enter the ISBN of the book you would like to add: 1234567890
The ISBN you searched was not found in the inventory.
A new record will be created for this ISBN

Enter wholesale price: 40.95

Enter quantity to be added: 5

Enter retail price: 55.99

Enter the day for date added: 7

Enter the month for date added: 12

Enter the year for date added: 2017

Enter the book title: Fundamentals of Quantum Mechanics

Enter the book author: Mary Lou Johnson

Enter the publisher: McGraw Hill

 New book record created.
The information for the book created is:

ISBN: 1234567890
Title: Fundamentals of Quantum Mechanics
Author: Mary Lou Johnson
Publisher: McGraw Hill
Wholesale Price: 40.95
Quantity on Hand: 5
Retail Price: 55.99
Month added: 12
Day added: 7
Year added: 2017
                 Serendipity BookSellers
                   Inventory Database
 1. Look up a Book
 2. Add a Book
 3. Edit a Book
 4. Delete a Book
 5. Return to the Main Menu
```

Z:\animegerdichian\Project\x64\Debug\Project.exe

```
                Serendipity BookSellers
                  Inventory Database
 1. Look up a Book
 2. Add a Book
 3. Edit a Book
 4. Delete a Book
 5. Return to the Main Menu

 Enter your Choice : 3
You selected edit book.
Enter the ISBN of the book you would like to edit: 1234567890

Enter wholesale price: 60.99

Enter quantity to be added: 4

Enter retail price: 45.99

Enter the day for date added: 7

Enter the month for date added: 12

Enter the year for date added: 2017

Enter the book title: Introduction to the Fundamentals of Quantum Mechanics

Enter the book author: Mary Lou Johnson

Enter the publisher: McGraw Hill

 Book has been edited.
The information for the book created is:

ISBN: 1234567890
Title: Introduction to the Fundamentals of Quantum Mechanics
Author: Mary Lou Johnson
Publisher: McGraw Hill
Wholesale Price: 60.99
Quantity on Hand: 4
Retail Price: 45.99
Month added: 12
Day added: 7
Year added: 2017
                Serendipity BookSellers
                  Inventory Database
 1. Look up a Book
 2. Add a Book
 3. Edit a Book
 4. Delete a Book
 5. Return to the Main Menu

 Enter your Choice : 4
You selected delete up book.
Enter the ISBN of the book you would like to delete: 1234567890
The information for the book selected is:

ISBN: 1234567890
Title: Introduction to the Fundamentals of Quantum Mechanics
Author: Mary Lou Johnson
Publisher: McGraw Hill
Wholesale Price: 60.99
Quantity on Hand: 4
Retail Price: 45.99
Month added: 12
Day added: 7
Year added: 2017
Are you sure you want to delete this item? N
No items deleted from inventory.
                Serendipity BookSellers
                  Inventory Database
 1. Look up a Book
 2. Add a Book
 3. Edit a Book
 4. Delete a Book
 5. Return to the Main Menu

 Enter your Choice : _
```

Z:\animegerdichian\Project\x64\Debug\Project.exe

```
Inventory accessed successfully.

                  Serendipity Booksellers
                         Main Menu
1. Cashier Module
2. Inventory Database Module
3. Report Module
4. Exit

Enter your choice: 3
                 Serendipity BookSellers
                         Reports
 1. Inventory listing
 2. Inventory Wholesale Value
 3. Inventory Retail Value
 4. Listed by Quantity
 5. Listed by Cost
 6. Listed by Age
 7. Return to Main Menu

Enter your choice: 2

 You selected report Wholesale Value


 Book Title                                          Quantity      WholesalePrice        ItemTotal

Textbook of Medical Physiology                           4              109.99              439.96
Starting Out with C++                                    5              175.99              879.95
Learning R                                               4               29.99              119.96
R for Data Science                                       5               35.99              179.95
Hands On Machine Learning with Scikit                    4               34.99              139.96
Introduction to Statistical Methods & Data Analysis      5              158.95              794.75
The little SAS Book                                      4               44.95              179.80
Learning SAS by Examples                                 5               75.95              379.75
R for Dummies                                            6               21.99              131.94
Lippincott's Pharmacology                                4               65.99              263.96
Remingtons's Pharmaceutical Sciences                     3              181.99              545.97
Practice & Problem Solving Geometry Workbook             6                7.85               47.10
Remington's Science & Practice of Pharmacy               4              172.99              691.96
Introduction to Porgramming with C++                     5              143.00              715.00
Teamwork and Project Management                          4               49.99              199.96
Calculus                                                 4               75.99              303.96
Engineering Fundamentals & Problem Solving               3              199.99              599.97
Basic Physics                                            5               17.99               89.95
Chemistry: The Central Science                           4              175.99              703.96
Chemistry: A Molecular Approach                          4              149.99              599.96
The Organic Chemistry of Drug Design                     4               79.99              319.96
Practical Process Research & Development                 2              127.99              255.98
Mass Spectrometry                                        3               34.99              104.97
Introduction to the Fundamentals of Quantum Mechanics    4               60.99              243.96

 The total Wholesale Value of Inventory is : $8932.64

                 Serendipity BookSellers
                         Reports
 1. Inventory listing
 2. Inventory Wholesale Value
 3. Inventory Retail Value
 4. Listed by Quantity
 5. Listed by Cost
 6. Listed by Age
 7. Return to Main Menu

Enter your choice: _
```

Z:\animegerdichian\Project\x64\Debug\Project.exe

```
                  Cashier Menu
 1. Add a book to Cart
 2. Edit Cart
 3. Checkout
 4. Return to the Main Menu

 Enter your Choice : 1

Enter ISBN: 0128041293

Enter the quantity needed: 5

Insufficient quantity in inventory.

We have 3 quantities on hand

Would you like to modify the quantity needed? (Y/N) : Y

Enter the new quantity needed: 1

Add another book to the order?(Y/N): N

                Serendipity BookSellers

                  Cashier Menu
 1. Add a book to Cart
 2. Edit Cart
 3. Checkout
 4. Return to the Main Menu

 Enter your Choice : 2

Enter ISBN: 0128041293

Your cart contains 1 copies of this book.
Enter the new quantity: 3

There are now 3 copies of 0128041293 in your cart.

                Serendipity BookSellers

                  Cashier Menu
 1. Add a book to Cart
 2. Edit Cart
 3. Checkout
 4. Return to the Main Menu

 Enter your Choice : 3

                Serendipity Booksellers

  Date: 12/7/2017

 Qty    ISBN           Title                                    Price          Total
-------------------------------------------------------------------------------------

 3      0128041293     Mass Spectrometry                        39.99          119.97



                       Subtotal       119.97
                       Tax            10.80
                       Total          130.77

  Thank you for shopping at Serendipity BookSellers!


                Serendipity BookSellers

                  Cashier Menu
 1. Add a book to Cart
 2. Edit Cart
 3. Checkout
 4. Return to the Main Menu

 Enter your Choice : _
```

Z:\animegerdichian\Project\x64\Debug\Project.exe

```
                Serendipity BookSellers
                        Reports
 1. Inventory listing
 2. Inventory Wholesale Value
 3. Inventory Retail Value
 4. Listed by Quantity
 5. Listed by Cost
 6. Listed by Age
 7. Return to Main Menu

Enter your choice: 6

 You selected report list by age.

 Enter the current year: 2017

 Enter the current month: 12

Book Title                                              Months on Shelf
Remingtons's Pharmaceutical Sciences                            43
Learning R                                                      40
Remington's Science & Practice of Pharmacy                      34
Starting Out with C++                                           33
Calculus                                                        33
The little SAS Book                                             32
Textbook of Medical Physiology                                  31
Teamwork and Project Management                                 20
Basic Physics                                                   20
Mass Spectrometry                                               19
Introduction to Porgramming with C++                            18
Hands On Machine Learning with Scikit                           18
Lippincott's Pharmacology                                       18
Introduction to Statistical Methods & Data Analysis             17
Learning SAS by Examples                                        17
Practice & Problem Solving Geometry Workbook                    10
Chemistry: The Central Science                                  10
R for Dummies                                                    9
Chemistry: A Molecular Approach                                  9
Engineering Fundamentals & Problem Solving                       8
The Organic Chemistry of Drug Design                             4
Practical Process Research & Development                         3
R for Data Science                                               3
Introduction to the Fundamentals of Quantum Mechanics            0

                Serendipity BookSellers
                        Reports
 1. Inventory listing
 2. Inventory Wholesale Value
 3. Inventory Retail Value
 4. Listed by Quantity
 5. Listed by Cost
 6. Listed by Age
 7. Return to Main Menu

Enter your choice: _
```

Ani Megerdichian
Darshana Palekar
Zhenning Guo

CIS22B Final Project Pseudocode

Cashier.h

include all the libraries and header files required
declare calss Cashier,inheritance with Module
private part in class:
declare numOrders as integer
declare count as integer
public part in calss:

default constructor of Cashier, takes 0 arguments:
set numOrders and count to 0
end of constructor
default destructor of Cashier, takes 0 arguments:
print out a message and notice the user destructor is initiated

declare void function setNumOrders,takes in 1 integer argument and returrn nothing
declare const function getNumOrders, takes 0 argument and return an integer

declare void function setCount, takes 1 integer and returrn nothing
declare function getCount, takes 0 arguments and return an ingeter

declare void function module,takes in 2 argument and return nothing, the first argument is the list of book name, the second argument is a constant which is the book size

declare function calcTotal, takes in 2 argument and return a double, the first one is the price of book as double, the second one is the count of number as an integer

declare function calcSubtotal,takes in 2 argument and return a double, the first one is the price list of book as double, the second one is the count of number as an integer

declare function calcTax, takes in 2 argument and return a double, the first one is the subtotal value as double, the second is the constant double tax rate

declare function calcOrderTotal, takes 2 arguments and return a double, the first one is the value of subtotal as double, the second one is the value of tax as double

declare function addOrder, takes 0 argument and return a character

declare void function addToCart, takes 6 arguments and return nothing. It takes book list from BookInfo class, a integer SIZE, integer list order indicates, integer list orderQuantity, integer count and integer numOrders

declare void function editCart takes 5 arguments and return nothing. It takes book list from BookInfo class, a integer SIZE, integer list indicates, integer list orderQuantity, and date from class Date

declare void function checkout takes 5 arguments and return nothing. It takes book list from BookInfo class, a integer SIZE, integer numOrders, integer list order indicates,integer list orderQuantity and date from Date class

end of class

Cashier.cpp

include all the libraries and header files required
set std as namespace

Cashier class void function setNumOrders,takes in 1 integer argument:
set numOrders equal to the argument taken

Cashier class const function getNumOrders, takes 0 argument and return an integer:
return the value of numOrders as integer

Cashier class void function setCount, takes 1 integer and returrn nothing:
assign the value of count to the taken argument

Cashier class function getCount, takes 0 arguments and return an ingeter
return the value of count

void function module,takes in 2 argument and return nothing, the first argument is the list of book name, the second argument is a constant which is the book size:
assign a constant integer ARRSIZE as 30
declare integer list orderIndices with size of ARRSIZE as a list with 0
declare integer list orderQuantity with size of ARRSIZE as a list with 0

prompt a message ask the user to enter the day of the book being added
user input day as bkDay
prompt a message ask the user to enter the month of the book being added
user input month as bkMon
prompt a message ask the user to enter the year of the book being added
user input year as bkYear

declare a Date class called date with the 3 variables bkDay,bkMon,bkYear

set integer menuchoice to 0
while loop, exits when menuchoice equals to 4

print out the basic cashier UI
ask the user to type in his choice
the user type in menuchoice value

when menuchoice less than 1 or larger than 4
prompt a error message and let user type again
the user type in menuchoice value
switch function takes menuchoice as argument
if menuchoice is 1:
call addToCart function by using the book list from BookInfo class, a integer SIZE, integer list order indicates, integer list orderQuantity, integer count and integer numOrders
break loop
if menuchoice is 2:
call editCarttakes function by using book list from BookInfo class, a integer SIZE, integer list indicates, integer list orderQuantity, and date from class Date
break loop
if menuchoice is 3:
call checkout function by using book list from BookInfo class, a integer SIZE, integer numOrders, integer list order indicates,integer list orderQuantity and date from Date class
break loop
if menuchoice is 4:
prompt a message telling the user he is exiting the module
break loop
other scenarios:
tell the user he need to enter a valid choice
end of function

void Cashier class function addToCart takes 6 arguments and return nothing. It takes book list from BookInfo class, a integer SIZE, integer list order indicates, integer list orderQuantity, integer count and integer numOrders:
declare integer transindex
declare string transISBN
declare integer transQuantity
declare integer neededQuantity
declare integer oIndex
declare character choice as "Y"
declare character option

do while loop:
ask the user to type in isbn number
call function searchForISBN by using isbn, book list and const SIZEa

if transIndex equals to -1:
tells the user book is not found
ask the user if he wants another try

user type his choice
if transIndex is not -1:
get number of quality through calling getQtyOnHand of book list, save it as transQuantity
get index number by calling findIndex, using transIndex, orderIndices and SIZE, save it as oIndex

if transIndex and orderQuantity with index of oIndex is not 0, or transIndex larger than 0 and oIndex is not -1:
tell the user the book is already in storage
let him use edit chart command to edit book info
ask him if he want another isbn
user type choice
discard cin value
else if transQuantity is 0:
tell the user book is out of stock
ask him if he want another isbn
user type choice
discard cin value

else if transQuantity is larger than 0:
ask the user how many book he want
user type choice
discard cin value

if neededQuantity is bigger than transQuantity:
tell the user book is not enough
and report how many book in storage
ask the user if he want to change his required copy
user type choice
discard cin value
if user type Y:
ask the user type new number
type neededQuantity

if neededQuantityi is less or equal to transQuantity:
increase numOrders by 1
orderIndices with index count equal to transIndex
orderQuantity with index count equal to neededQuantity
increase count by 1
subtract the stored copy of book by the required nubmer

reset transQuantity to0
call addOrder function and let the returned value equal to choice

do while loop exit condition: choice equals to Y or y

call setcount function by using count as argument

void Cashier function editCart.It takes book list from BookInfo class, a integer SIZE, integer list indicates, integer list orderQuantity, and date from class Date

declare string transISBN
declare integer transIndex
declare integer appqty
declare integer appindex
declare character option
declare integer origqty

ask the user to type in isbn
user type in isbn
clear cin

store bookindex by calling searchForISBN, using transisbn, book list and SIZE
store appindex by calling findIndex, using transIndex, orderIndices and SIZE

if transIndex equal to -1 or appindex equal to -1 or transIndex not equal to -1 or appindex not equal to -1 and orderQuantity with index appindex equal to 0:

print out "This ISBN was not found in your cart. Would you like to try again?(Y/N)"
input option
clear input
if option equal to 'Y' or option equal to 'y':
print "Enter the ISBN: "
input transISBN
clear input
call searchForISBN use transISBN, book, SIZE as argument, the returnning value equal to transIndex
call findIndex using transIndex, orderIndices, SIZE,the returnning value equal to appindex
if transIndex is not -1 and appindex is not -1:
orderQuantity with index appindex equal to origqty
print "Your cart contains "and orderQuantity with index appindex," copies of this book."
print "Enter the new quantity: "
input appqty
if book with index transIndex calling getQtyOnHand plus orderQuantity with index appindex is less than appqty:
print "Insufficient quantity in inventory."
print "We have " ,book with index transIndex calling getQtyOnHan, " more quantities on hand"
print "\nWould you like to modify the quantity needed? (Y/N) : "
input option
Clear input

```
if option equal to  'Y':
print "Enter the new quantity: "
input appqty
if book with index transIndex calling function getQtyOnHand plus  orderQuantity with index
appindex] is not less than appqty:
book with index transIndex calling function setQtyOnHand with argument origqty plus book
with index transIndex calling function getQtyOnHand minus appqty
orderQuantity with appindex as index equal to appqty
print "There are now " ,orderQuantity with index appindex " copies of " transISBN " in your
cart."
```

void Cashier function checkout, takes 5 arguments and return nothing. It takes book list from BookInfo class, a integer SIZE, integer numOrders, integer list order indicates,integer list orderQuantity and date from Date class:

```
set constant integer ARRSIZE as 30
declare double list with size of ARRSIZE
declare double orderTotal as 0.0
declare double tax as 0.0
declare double subtotal as 0.0
declare double sales_tax as 0.09


if numOrders is not 0 :
for integer i equal to 0 and i less than numOrders i increase by 1 each loop
if orderQuantity with index i is not 0:
orderTotals with index i equal to result of function calcTotal with argument book with index
orderIndices with index i as function getRetailPrice, orderQuantity with index i]
subtotal equal to calcSubtotal with argument orderTotals, numOrders
tax equal to return value of calcTax with argument subtotal, sales_tax
orderTotal equal to return value of calcOrderTotal with argument subtotal, tax
print "Serendipity Booksellers"
print " Date: " ,date
print "Qty ISBN Title      Price    Total"
print"_____
_____"
for integer i equal to 0, i less than numOrders iincreased by 1 each loop
if orderQuantity with index i is not 0:
print " ", orderQuantity with index i
print left , and setw with index 14, book with index orderIndices with index i and call function
getisbn
print left, setw with index 55, book with index orderIndices with index i call function getTitle
print fixed, showpoint, right, setprecision as 2
print setw with index 10, book with index orderIndices with index i calling function
getRetailPrice
```

print setw with index 10, orderTotals with index i
orderIndices with index i equal to 0
orderQuantity with index i equal to 0
print white spaces
print " Subtotal"
print left, setw with index 6, subtotal
print "Tax"
print left, setw with index 6, tax
print "Total"
print left, setw with index6, orderTotal
print "Thank you for shopping at Serendipity BookSellers!"
Call setNumOrders with 0
Call setCount with 0

Cashier functionaddOrder take 0 argument return a character
character option
print "Add another book to the order?(Y/N): "
input option
return option

Cashier functioncalcTotal take double price, integer count as argument return double
double total equal to count times price
return total

Cashier functioncalcSubtotal take double list total, int count as argument and return double
double subtotal equal to 0.0
for integer i equal to 0 i less than count i increase by 1 each loop
subtotal +equal to total with index of i
return subtotal

Cashier function calcTax take double subtotal, double salestax as argument return double:
double tax equal to subtotal times salestax
return tax

Cashier function calcOrderTotal take double subtotal, double tax as argument return double:
double orderTotal equal to subtotal plue tax
return orderTotal

Inventory.h

include all the libraries and header files required
declare class Inventory,inheritance with Module
public part in class

Inventory default constructor
Inventory default destructor
declare void function module that takes 2 parameters : BookInfo array, and array size
declare void function lookUpBook that takes 2 parameters : BookInfo array, and array size
declare void function editBook that takes 2 parameters : BookInfo array, and array size
declare void function addBook that takes 2 parameters : BookInfo array, and array size
declare void function deleteBook that takes 2 parameters : BookInfo array, and array size

Inventory.cpp

Define default Inventory constructor

Define Inventory destructor

void function Inventory takes two paramenters, BookInfo array and array size

set integer menuchoice to 0
while loop, exits when menuchoice equals to 5
print out the basic inventory UI
ask the user to type in his choice
the user type in menuchoice value

when menuchoice less than 1 or larger than 5
prompt an error message and let user type again
the user type in menuchoice value
switch function takes menuchoice as argument
switch (choice)

if menu choice is 1:
function lookUpBook takes two parameters, BookInfo object array and size of array
break;
if menu choice is 2:
function addBook takes two parameters, BookInfo object array and size of array
break;

if menu choice is 3:
function editBook takes two parameters, BookInfo object array and size of array
break;

if menu choice is 4:
function deleteBook takes two parameters, BookInfo object array and size of array
break;

if menu choice is 5:
return to main menu

void function lookUpBook(BookInfo book[], int SIZE)

declare variable searchedISBN and searchedIndex
ask user for searchedISBN
use searchForISBN fxn to find index of searchedISBN
if ISBN not found, display message
else, print book info

void function addBook(BookInfo book[], int SIZE)

declare variables for all BookInfo object attributes
declare variables for searchedISBN, searchedIndex, and addedCopies

get the ISBN of the book from the user
search for ISBN using searchForISBN fxn

if found, display message
ask user the number of copies they would like to add
set the number of copies+ qtOnHand for specific book as the new qtyOnHand
ask for date
set date values for specific book
display message

if not found, display message
ask user for attributes for new book
use BookInfo default constructor to create new book in BookInfo array

else
display message saying a book cannot be added at the time

void function editBook(BookInfo book[], int SIZE)

declare variables for all BookInfo object attributes

declare variables for searchedISBN, searchedIndex

get the ISBN of the book from the user
search for ISBN using searchForISBN fxn

if found, display message saying book was found
if not found, display message saying book was not found

if found, ask user for attributes for new book
use BookInfo default constructor to create new book in BookInfo array
print attributes of book

void function deleteBook(BookInfo book[], int SIZE)

declare variables for all BookInfo object attributes
declare variables for searchedISBN, searchedIndex, and confirm

char confirm;

cout << "Enter the ISBN of the book you would like to delete: ";
cin >> searchedISBN;

get the ISBN of the book from the user
search for ISBN using searchForISBN fxn

if not found, display message

if found, validate delete choice, and then delete if confirmed

Report.h
Include all the required header files
Declare class Reports, inheritance with class Module
Public part:
Default constructor
Default destructor
declare void function module with argument BookInfo class list book, int SIZE
declare void function reportInventoryList with argument BookInfo class list book, int SIZE
declare void function reportWholesaleValue with argument BookInfo class list book, int SIZE
declare void function reportRetailValue with argument BookInfo class list book, int SIZE

declare void function reportListByQuantity with argument BookInfo class list book, int SIZE
declare void function reportListByCost with argument BookInfo class list book, int SIZE
declare void function reportListByAge with argument BookInfo class list book, int SIZE
declare void function selectionSort2descending with argument integer list array1, string array2[], int size
declare void function selectionSort2descending with argument double list array1, string list array2, int size
end of class

Report.cpp

Include all the required libraries and header files

using std as namespace

default constructor of class Reports
default destructor of class Reports

assign const integer ARR_SIZE equal to 30

void Reports class functionmodule with argument BookInfo class list book, integer SIZE

integer choice equal to 0
while choice not equal to 7
print " Serendipity BookSellers "
print " Reports "
print " 1. Inventory listing "
print " 2. Inventory Wholesale Value "
print " 3. Inventory Retail Value "
print " 4. Listed by Quantity "
print " 5. Listed by Cost "
print " 6. Listed by Age "
print " 7. Return to Main Menu "
print "Enter your choice: "
input choice
while choice less than 1 or choice larger than 7
print "Please enter a valid choice in the range 1 - 7. "
print "Enter your choice: "
input choice
switch argument with choice
case 1:
call reportInventoryList with argument book and size
break
case 2:
call reportWholesaleValue with argument book and size

break
case 3:
call reportRetailValue with argument book and size
break
case 4:
call reportListByQuantity with argument book and size
break
case 5:
call reportListByCost with argument book and size
break
case 6:
call reportListByAge with argument book and size
break
case 7:

void Call reports class functioncall reportInventoryList with argument BookInfo class list book, integer SIZE
print "You selected call report inventory list. "
print " Serendipity Booksellers "
print " Inventory List Call report"
print "ISBNBook Title Author Publisher DateAdded"
print " WholesalePrice Quantity RetailPrice "
for loop as integer index equal to 0 index less than SIZE index++
if book with index index and getisbn not equal to ""
print left , print isbn of the book
print left , print title of the book
print left , print author of the book
print left , print publisher of the book
Date class date1book[index].getDay, book[index].getMonth, book[index].getYear
Print formatted Output

void Call reports class functioncall reportWholesaleValue with argument book and size
print " You selected call report Wholesale Value "
double itemTotal list ARR_SIZE
double total equal to 0
for integer index equal to 0 index less than SIZE index
itemTotal with index index equal to 0
if book[index].getisbn not equal to ""
itemTotal[index] equal to book[index].getQtyOnHand times book[index].getWholesalePricetotal +equal to itemTotal[index]
print " Book Title Quantity WholesalePrice ItemTotal"
for integer index equal to 0 index less than SIZE index++

if book[index].getisbn not equal to ""

```
Print formatted Output
print " The total Wholesale Value of Inventory is : $" , total , endl
print endl
void Call reports class function with argument call reportRetailValueBookInfo book[], integer
SIZE
print " You selected call report retail value."
double itemTotal[ARR_SIZE]
double total equal to 0
for integer index equal to 0 index less than SIZE index++
itemTotal[index] equal to 0
if book[index].getisbn not equal to ""
itemTotal[index] equal to book[index].getQtyOnHand times book[index].getRetailPrice
total +equal to itemTotal[index]
print "Book Title        QuantityRetailPrice   ItemTotal"
for integer index equal to 0 index less than SIZE index++
if book[index].getisbn not equal to ""
Print formatted Output
print " The total Retail Value of Inventory is $: " , total , endl
print endl
void Call reports class functioncall reportListByQuantityBookInfo book[], integer SIZE
print " You selected call report list by quantity. "
string bookarrTitle[ARR_SIZE]
integer bookarrQty[ARR_SIZE]
for integer index equal to 0 index less than SIZE index++
bookarrTitle[index] equal to book[index].getTitle
bookarrQty[index] equal to book[index].getQtyOnHand
selectionSort2descendingbookarrQty, bookarrTitle, SIZE
print "Book Title" , "QuantityOnHand"
for integer index equal to 0 index less than SIZE index++
print endl
if book[index].getisbn not equal to ""
Print formatted Output
print endl
void Call reports class functioncall reportListByCostBookInfo book[], integer SIZE
print " You selected call report list by cost."
string bookarrTitle[ARR_SIZE]
double bookarrWPrice[ARR_SIZE]
for integer index equal to 0 index less than SIZE index++
bookarrTitle[index] equal to book[index].getTitle
bookarrWPrice[index] equal to book[index].getWholesalePrice
selectionSort2descendingbookarrWPrice, bookarrTitle, SIZE
print "Book Title" , "WholesalePrice"
for integer index equal to 0 index less than SIZE index++
if book[index].getisbn not equal to ""
```

```
Print formatted Output
print endl
void Call reports class functioncall reportListByAgeBookInfo book[], integer SIZE
print " You selected call report list by age."
integer bookarrAge[ARR_SIZE]
string bookarrTitle[ARR_SIZE]
integer monththen, monthnow, yearthen, yearnow
integer age, agemonths, ageyears
print " Enter the current year: "
input yearnow
print " Enter the current month: "
input monthnow
for integer index equal to 0 index less than SIZE index++
if book[index].getTitle not equal to ""
bookarrTitle[index] equal to book[index].getTitle
monththen equal to book[index].getMonth
yearthen equal to book[index].getYear
if monththen less than monthnow && yearthen less than yearnow
ageyears equal to yearnow - yearthen
agemonths equal to monthnow - monththen
bookarrAge[index] equal to 12 times ageyears + agemonths
else if monththen less than monthnow && yearthen equal toequal to yearnow
agemonths equal to monthnow - monththen
bookarrAge[index] equal to agemonths
else
ageyears equal to yearnow - yearthen - 1
agemonths equal to 12 - monththen - monthnow
bookarrAge[index] equal to 12 times ageyears + agemonths
selectionSort2descendingbookarrAge, bookarrTitle, SIZE
print "Book Title" , " Months on Shelf"
for integer index equal to 0 index less than SIZE index++
if book[index].getWholesalePrice not equal to 0.0
Print formatted Output
void Call reports class functionselectionSort2descendinginteger array1[], string array2[], integer
size
integer startScan, minIndex, maxValue1
string maxValue2
for startScan equal to 0 startScan less than size - 1 startScan++

minIndex equal to startScan
maxValue1 equal to array1[startScan]
maxValue2 equal to array2[startScan]
for integer index equal to startScan + 1 index less than size index++
if array1[index] larger than maxValue1
maxValue1 equal to array1[index]
```

minIndex equal to index
maxValue2 equal to array2[index]
array1[minIndex] equal to array1[startScan]
array1[startScan] equal to maxValue1
array2[minIndex] equal to array2[startScan]
array2[startScan] equal to maxValue2

void Call reports class functionselectionSort2descendingdouble array1[], string array2[], integer size
integer startScan, minIndex
double maxValue1
string maxValue2
for startScan equal to 0 startScan less than size - 1 startScan++
minIndex equal to startScan
maxValue1 equal to array1[startScan]
maxValue2 equal to array2[startScan]
for integer index equal to startScan + 1 index less than size index++
if array1[index] larger than maxValue1
maxValue1 equal to array1[index]
minIndex equal to index
maxValue2 equal to array2[index]
array1[minIndex] equal to array1[startScan]
array1[startScan] equal to maxValue1
array2[minIndex] equal to array2[startScan]
array2[startScan] equal to maxValue2

Date.h

Date class
    private:
        day, month, year
    public:
        default constructor, second constructor, destructor
        setters and getters
        overloaded insertion operator

Date.cpp

First constructor
    set default values for member variables
Second constructor

assign parameters of constructor to member variables
Destructor
Overloaded insertion operator
takes reference of Date object as parameter
returns day, month, and year of object in formatted manner
Setters and Getters

BookData.h

BookData class
private:
isbn, wholesalePrice, qtyOnHand, retailPrice
public:
default constructor, second constructor, destructor
getters
setQtyOnHand

BookData.cpp

First constructor
set default values for member variables
Second constructor
assign parameters of constructor to member variables
Destructor
setQtyOnHand
Getters

BookInfo.h

BookInfo class extends BookData and Date

private:
title, author, publisher
public:
default constructor, second constructor, destructor
overloaded insertion operator
getters

BookInfo.cpp

First constructor
set default values for member variables
Second constructor
assign parameters of constructor to member variables

Destructor
Overloaded insertion operator
    takes reference of BookInfo object as parameter
    returns BookInfo object attributes separated by commas
Getters

templatefxns.h

Type findIndex(Type var, Type array, Type array size)
    initialize bool variable as false
    initialize int index variable to 0

    while var is not found and index is less than size
        if value of array at index == var
            bool variable = true
        else
            increment index

    if found
        return index
    else
        return -1

MainMenu.cpp

Define a constant to hold array size

Open inventory file
If file opens successfully, display message
Else, display error message

Declare array of book objects
Declare variables for object member variables

Read file line by line
For i = 0, i < number of attributes(tokens), I++
    put read tokens in an array
    convert strings to integer if needed

Populate book objects using overloaded constructor

Create Cashier, Reports, and Inventory objects

Create Module pointers to point to these objects

Initialize variable for user menu choice

Display Main Menu
1. Cashier
2. Inventory
3. Report
4. Exit

While menu choice is not valid
ask for choice again

Switch (menu choice)
case 1:
cashierpointer->module fxn;
break;
case 2:
inventorypointer->module fxn;
break;

case 3:
reportspointer->module fxn;
break;

case 4:;

Close file

Open temporary file
For i = 0, i < size of array, i++
if book isbn is not empty
use overloaded << operator to write book to file

Close file

Delete old file

Rename new file

Searchfor.cpp

searchForISBN fxn

```
Initialize bool found variable to false
Initialize int index variable to 0

While not found and index is less than array size
        If book isbn == isbn searched for
                Found = true
        Else
                Increment index

If found
        Return index
Else
        Return -1


getIndex fxn

Initialize bool found variable to false
Initialize int index variable to 0

While not found and index is less than array size
        If book isbn == empty
                Found = true
        Else
                Increment index

Return index;
```