name: main

return value: zero

parameter(s): none

**purpose: delegate tasks for the sale of books to different functions**

The list of books is retrieved from the book file and the number of available books is determined by the retrieveBooks function. The list of students is retrieved from the student file.

The choice selected in the displayMenu module is then used. While the choice selected in the displayMenu module is not "MAIN\_EXIT" which represents the integer value five (5), a set of tasks are done depending on the choice.

If the choice is "READ", the integer value one (1), the student list and the number of students are passed to the addStudent module to add the student to the list and update the number of students, if necessary. The information about new book is then read in the readBook module which also updates the number of books available.

If the choice is "SELL", the integer value two (2), the list of books and students, and the number of books and students are passed to the makeSale module so tasks regarding the sale of a book can occur. This module also updates the number of books available.

If the choice is “DELETE\_STUDENT”, the integer value three (3), the user is prompted to enter the last 4 digits of the student ID. After this is read, it and the list of students and books and the number of students and books are passed to the deleteStudent module to remove the indicated student from the student list and her books from the book list.

Lastly, if the choice is "PRINT\_REPORT", the integer value four (4), the list of books and students, and the number of books and students are passed to the reportMain module so tasks regarding the generation of reports can occur.

Another choice is then determined and this repeats until the choice is "MAIN\_EXIT", the integer value five (5). Afterwards, the list of books and the number of books is passed to the saveBooksmodule so the books are rewritten to the book file, likewise for the saveStudentsmodule.

name: displayMenu

return value: integer representing user's choice

parameter(s): none

purpose: presents a list of choices to the user

A list of options is presented then the user is prompted to enter a choice. The integer value representing the choice is read. If the value is less than one, which represents new book donation, or greater than 5, which represents exiting the program, an error statement is displayed and the user is prompted to enter a choice again until a valid choice is entered. The choice is then returned to the main module.

name: retrieveBooks

return value: the number of current available books

parameter(s): an empty list of books

purpose: update book count and populates the book list from file

The book list file is opened. If it is opened successfully, while data is being read from the file, the title, serial number, author, subject, price, and the ID of the owner for all the books are stored in an array. After each book is read from file, the number of books is incremented. The number of books is then returned to the main module.

name: retrieveStudents

return value: the number of students

parameter(s): an empty list of students

purpose: update student count and populates the student list from file

The student list file is opened. If it is opened successfully, while data is being read from the file, the ID, full name, grade and money owed for all the students are stored in an array. After each student is read from file, the number of students is incremented. The number of students is then returned to the main module.

name: checkStudent

return value: integer value representing whether the student is new or returning

parameter(s): list of students, the number of students, the ID number of a student

purpose: checks if student is new

The ID of the student contributing a new book is checked against the IDs of list of students to determine whether the student is a new or returning student. If the student is new, the integer value returned to the calling module is 0, which represents "yes". Otherwise, the value returned is 1, which represents "no".

name: addStudent

return value: the number of students

parameter(s): list of students, the number of students, placeholder for student ID to be read

purpose: information about new student is read

The user is prompted to enter the last 4 digits of the student ID. After this is read, it is passed, along with the student list, and the number of students, to checkStudent module to determine if the student is new or returning. If the student is new, then the new student is added to the student list array. The user is prompted to enter the name of the student and this information is read. The money owed to the student is initialised to zero. The number of students is incremented and then returned to the main module.

name: readBook

return value: the number of books

parameter(s): list of books, number of books, ID of owner of new book

purpose: reads information about new book

The user is prompted to enter the title, name of author, selling price and subject category of the new book. If the entered price is less than or equal to zero, an error message is displayed and the user is prompted to re-enter the price until a valid price is read. These details and the student ID which was passed to the module is added to the book list array. The number of books is incremented and returned to the calling module.

name: saveBooks

return value: none

parameter(s): list of books, number of books

purpose: rewrites the list of current books to book file

A file to store the information about all books is opened or created. If this is done successfully, the title, serial number, author, subject, price, and ID number of owner for each book is written to the file. A message confirming that the action has been performed is then displayed.

name: saveStudents

return value: none

parameter(s): list of students, number of students

purpose: rewrites the list of current students to student file

A file to store the information about all students is opened or created. If this is done successfully, the ID number, name, grade and total money owed for each student is written to the file. A message confirming that the action has been performed is then displayed.

name: makeSale

return value: number of books

parameter(s): list of books, number of books, list of students, number of students

purpose: available books are identified and the book and student lists are updated if book is sold

If the desired book is available but the user did not select any of those displayed, the user is asked if he/she wants of the options presented. If the user enters "yes", the user is then prompted to enter the serial number of the book chosen. Afterwards, a message indicating the sale of a book is displayed. If a book was available and selected to be bought, the number of books is updated by the deleteBook module. The user is then asked if he/she wants more books and prompted for choice. If the choice is "yes", then all previous procedures are repeated until the choice is "no". The number of books is then returned to the main module.

name: displayBooks

return value: none

parameter(s): list of books, number of books, the availability status of the book passed by reference, the sale status of book passed by reference, placeholder for bookID to be entered

purpose: available subject books are displayed

The availability and sale status of book is to be identified is set to "NO" which represents the integer value one (1).

The user is prompted to enter the subject of the wanted book. The number of books in the subject category is determined and a subject book list is populated by the readSubject module. If there are no books in that subject category, a message indicating this is displayed. Otherwise, the user is prompted for the title of the book wanted.

While the list is not empty, a book is removed from the front of the subject list. If its title matches, then the availability status of the book is set to "YES" which represents zero (0). The information about the book, excluding the ID of the owner and the date received, is displayed. The user is then asked if he/she wants that particular book. If the choice is "yes" then the sale status is changed to "YES", the serial number is placed in the bookID variable, and the function ends. Otherwise, the next book is searched for in the list and the previous procedures are repeated until no more matching books are found in the list. When this happens or if a matching book was never found, a message indicating that there are no more options available is displayed.

name: readSubject

return value: number of books relevant to a particular subject

parameter(s): empty list of subject books, number of subject books, list of books, number of books, the subject of wanted book

purpose: populates subject book list with relevant books

The number of books related to the subject is initialised to zero. The general book list is browsed through. If the book has a matching subject, the current subject list, number of subject books and the matching book information is passed to the enqueue module so it is added to subject list and the number of books for that subject is incremented. The number of subject books is returned to the calling module.

name: deleteBook

return value: the number of books

parameter(s): list of students and books, number of books and students, ID of book sold

purpose: deletes book sold from array of current books

The serial number of the sold book and the serial numbers of the books in the book list are compared. The array position of the matching book is stored in an integer variable. The book list from that index is then rewritten so the sold book is deleted from the list. The number of books is decremented and returned to the calling module.

name: calcTotal

return value: none

parameter(s): list of students, the number of students, the information about the book sold

purpose: price of the book sold is added to the student owner's total amount owed

The owner of the sold book is found and then owner's total money owed is increased by the book's price.

name: printTotalOwed

return value: none

parameter(s): list of students, the number of students

purpose: list of students who are to be paid are placed in file

The date of the file to be created is determined and formatted using various time functions. The date is stored in a string variable to be attached to the file name. A file that lists the amount of money owed to each student is opened or created. If this is done successfully, for all students whose total owed is greater than zero, their information is written to the file and their total owed amounts are reset to zero. A message confirming that the action has been performed is then displayed.

name: displayReportMenu

return value: integer representing user's report choice

parameter(s): none

purpose: presents a list of report choices to the user

A list of report options is presented then the user is prompted to enter a choice. The integer value representing the choice then is read.

If the value is less than one, which represents generating the student payment list, or greater than 4, which represents returning to the main menu, an error statement is displayed, and the user is prompted to re-enter a choice until a valid choice is entered.

The choice is then returned to the report main module.

name: findStudent

return value: position of student in array

parameter(s): list of students, number of students, ID of student to be removed from list

purpose: identifies the array index of the student to be removed from student list

The ID number of the student to be removed and the ID numbers of the students in the list are compared. The array position of the matching student is stored in an integer variable and returned to the calling module.  If the student is not found, then an error message is displayed and the value negative one (–1) is returned.

name: reportMain

return value: none

parameter(s): list of students, number of students, list of books, number of books

purpose: report function driver

 A variable representing copy of the book list is declared; afterwards, it is populated using the loadBooks module and sorted by the sortBooks module.

 The choice selected in the displayReportMenu module is then used. While the choice selected in the displayMenu module is not "EXIT" which represents the integer value four(4),  a report is generated depending on the choice.

 If the choice is "PAYMENT", the integer value one(1), the student list and the number of students are passed to printTotalOwed module so the students who are owed money are written to a file.

 If the choice is " SUBJ\_COUNT", the integer value two(2), the sorted book list and the number of books are passed to the displaySubjCount module so the current subject categories and the quantity of books under each category are written to a file.

 Lastly, if the choice is " SUBJ\_BOOK\_COUNT", the integer value three (3), the sorted book list and number of books are passed to the displaySubjBooks module so the available books under a specified subject and their quantity are written to a file.

This repeats until the choice is "EXIT", the integer value four (4). Afterwards, the user is returned to the main menu.

name: sortBooks

return value: none

parameter(s): copy of book list, number of books

purpose: sorts the book list alphabetically by subject and then book title

First, the book list is sorted alphabetically using a selection sort. The subject of the book at each array index is compared with the remaining in the list (subarray). During each pass, it is assumed that the book at the beginning of the subarray being sorted is to be the first listed, so it’s position is stored in an integer variable. This assumption is tested during the comparisons with the remaining books. If a book is found to come before the assumed first listed book alphabetically, its position is stored as the first listed book. This is repeated until there are no more books to compare in the list. Then, if the position of the first listed book was changed, the positions of the assumed first listed book is swapped with the actual first listed book. Afterwards, the list is sorted alphabetically by title using another selection sort.

name: loadBooks

return value: none

parameter(s): empty copy of book list, list of books, number of books

purpose: populates the book list to be sorted

The original book array list is copied to an empty book array list.

name: isFull

return value: result of comparison between the current size and maximum size of the subject book list

parameter(s): number of subject books, maximum number of books in list

purpose: compares the current size and maximum size of the subject book list to determine if the list is full

It compares the current number of subject books and the total book count and returns the comparison truth value.

name: isEmpty

return value: result of comparison between the current size and queue front index

parameter(s): number of subject books

purpose: compares the current size and queue front index to determine if the list is empty

It compares the current number of subject books and the queue front index (0) and returns the comparison truth value.

name: enqueue

return value: number of books relevant to a particular subject

parameter(s): current list of subject books, number of subject books, book to be added to subject list

purpose: adds book with matching subject to the current subject book list and updates the number of books in the list

The subject book identified in the readSubject module is added to the rear of the subject book queue and the current number of subject books is incremented. The updated number is returned to the calling module.

name: dequeue

return value: next book to be displayed to user

parameter(s): current list of subject books, number of subject books

purpose: removes the book from the subject book list and updates the number of books in the list

A variable to store the book item is declared and the subject book at the front of the subject book queue is assigned to it. Then, the number of subject books, passed by reference, is decremented. Afterwards, the remaining books in the queue is shifted downwards to remove the book from the list. The removed book is returned to displayBooks.

name: displaySubjCount

return value: none

parameter(s): sorted book list, number of books

purpose: displays each book subject category and number of books under the categories

The date of the file to be created is determined and formatted using various time functions. The date is stored in a string variable to be attached to the file name. A file that lists the available subjects and their book quantity is opened or created. If this is done successfully, the name of the subject is written to the file. The book quantity is determined using readSubject and stored in an integer variable. The quantity then is written is written to file. This process is repeated for the remaining subjects. A message confirming the report availability is then displayed.

name: displaySubjBooks

return value: none

parameter(s): sorted book list, number of books

purpose: displays the names and quantities of books under a specific subject category

The user is prompted to enter the name of the subject category to be examined. The sorted book list, an empty subject book list, the number of books, the indicated subject, and the subject book counter is passed to readSubject to populate the subject book list and determine the number of books under that subject category.

If books under the subject category are found, then the date of the file to be created is determined and formatted using various time functions. The date is stored in a string variable to be attached to the file name. A file that lists the available books and their quantity for the identified subject is opened or created. The subject book list is traversed and while the titles of the books in the list match, the book count for that book is incremented until a different book is identified. The subject, title and quantity of each book are written to file. A message confirming the report availability is then displayed.

If the subject is non-existent, an error message is displayed to the user.

name: deleteStudent

return value: none

parameter(s): list of students and books, number of books and students

purpose: deletes student from array of current students and updates the book record

The user is prompted to enter the last 4 digits of the student ID of the student to be removed from the list. The findStudent module is then called to determine the array position of the student in the list. If the student is found, the total owed and the unsold books owned by the student are displayed.

The book and student record are updated as the student is removed from the array list.