Mahinur Islam

M00912138

Project Report

Project Introduction:

The program is a library management system where the end users will be librarians, who can carry out five main functionalities: adding members, issuing books, returning books, calculating fines for overdue books, and displaying borrowed books. It employs object-oriented principles with classes such as Person, Librarian, Member, and Book, as well as a date class utilized to aid the calculation of the fines.

Classes:

* Person –

Attributes:

* name
* address
* email

Functions:

For every attribute (getName(), setName(), getAddress(), setAddress(), getEmail(), setEmail()), there are getter and setter functions available. These methods enable changing and accessing the Person class's private properties.

Constructors:

The default Initialises the Person object with default values. Parameterized Constructor: Sets the Person object's name, address, and email using the supplied values.

* Member –

Attributes:

* memberID
* bookLoaned

Functions:

* getMemberID() – Gets a member’s ID
* getBooksBorrowed() – Gets the book IDs a given member borrowed
* setBooksBorrowed(bookID) – The given ID is added to the list of borrowed books
* returnBook(int bookID) - The given ID is removed from the list of borrowed books ShowDetail() - Outputs member details while utilising person class

Constructors:

The class has a default constructor and a parameterized constructor that initializes the Person base class and sets the memberID.

* Librarian:

Attributes:

* staffID
* salary

Functions:

* addMember() - Adds members to the library system, prompting the input of member details, creating a Member object.
* issueBook(memberID, bookID) - Issues books to members and updates the member's list of borrowed books, while setting a due date.
* returnBook(memberID, bookID) - Handles the return of a borrowed book, calculating fines based on due date and returned date.
* listBooksBorrowed(memberID) - Lists all books borrowed by the given member
* Staff Salary and Staff ID getter and setter functions.
* readBooksFromCSV(int, books) - Reads data from the csv file.

Constructors:

The class has a default constructor and a parameterized constructor that initializes the Person base class and sets the staffID and salary attributes.

* Book :

Attributes:

* bookID
* bookName
* authorFirstName
* authorLastName
* bookType
* dueDate
* Member\* borrower

Functions:

The class's private attributes can be accessed by many getter functions, including getBookID(), getBookName(), getAuthorFirstName(), getAuthorLastName(), and getDueDate(). Only one setter, though, has setDueDate(Date date): uses the supplied Date object to set the book's due date. Lastly, there are the member functions. returnBook (memberID, int): uses the member ID as a parameter to determine whether the specified member borrowed the book. If so, the due date is removed, and the borrower pointer is reset to nullptr. It prints an error message if it does not. BorrowBook(Member\* member, Date date) is another member function that accepts a Date object as an argument and a Member pointer as input. establishes the due date and assigns the borrower to the specified member.

Constructors:

The constructor with the class is default and does not accept any arguments.

Additionally, it contains a parameterized constructor that initialises the corresponding attributes based on an ID, name, first and last name of the authors, and the type of book.

* Date –

Attributes:

* day
* month
* year:

All Private data representing the day, month, and year of the date.

Functions:

The class's private attributes can be accessed using the three getter functions (getDay(), getMonth(), and getYear()). Because these functions do not change the object's state, they are designated as const. Additionally, there is the date reseed method removeDate(): After essentially resetting the date by setting the day, month, and year to 0, the user input function getUserInputDate() asks the user to provide a date by providing the day, month, and year. These values are read by it from the console's standard input.

Constructors:

The class comes with a default constructor that initialises the date components without taking any parameters. Additionally, it provides a parameterized constructor that initialises the associated attributes by accepting integer values for day, month, and year.

Implementation of code in main.cpp file:

Header Inclusion:

The main.cpp file includes several header files (Person.hpp, Date.hpp, Member.hpp, Book.hpp, Librarian.hpp) containing declarations of classes and functions, which are utilised in the code in this file. (It also imports namespace std)

Main Function:

A librarian instance is created and book information is read from a CSV file using librarian.readBooksFromCSV(books) (assuming the books map is declared somewhere). Afterwards, by utilising a while loop the menu is displayed using ShowMenu() and the system processes user input until the user chooses to exit. User input is read as a string, converted to an integer using std namespace, and processed within a try-catch block to handle potential exceptions (invalid input or out-of-range).

By using a switch statemen different menu options are handled:

- 1: Adds a new member using librarian.addMember().

- 2: Issues a book to a member by taking member and book IDs as input.

- 3: Returns a book by taking member and book IDs as input.

- 4: Lists books borrowed by a member by taking member ID as input.

- 5: Exits the program.

The while loop continues the librarian exits.

The project’s main.cpp code provides a primary interactive interface for managing members and books in a library. However, functionality heavily relies on the implementation of the Librarian class as well as other related classes. Finally user validation is used to gracefully handle invalid credentials.

Testing approach:

The test code uses the Catch2 testing framework to write unit tests. The testing covers functionalities related to reading a CSV file, creating a new member, issuing books, listing books, setting and retrieving salary, and setting and retrieving staff ID. In total there are 6 tests:

Test cases:

* Reading CSV File

This test is checks if the CSV file reading function in the Librarian class works as expected.

Making use of a REQUIRE macro to ensure that the result of librarian.readBooksFromCSV(books) is "File successfully read library\_books.csv."

* Creating New Member

Testing that the ShowDetail function in the Member class correctly formats member details.

Creates two Member objects with different details and adds them to the members map.

Uses the REQUIRE macro to compare the output of member.ShowDetail() and member2. ShowDetail() with the desired strings.

* Issuing Book

This test verifies that the correct messages are printed when books are issued to a member. Calls setBooksBorrowed twice on a Member object. REQUIRE macro to check if the captured output matches the wanted string.

* List Books

A test verifying that the listBooksBorrowed function in the Librarian class correctly lists borrowed books. Calls librarian.listBooksBorrowed(1) to list books borrowed by a member. Uses the REQUIRE macro to check if the captured output matches the expected string.

* Salary

Testing the correctness of the SetSalary and GetSalary functions in the Librarian class.

Sets the salary of the Librarian using librarian.SetSalary(20000). With the REQUIRE macro program checks if the retrieved salary using librarian.GetSalary() is equal to the value, 20000.

* StaffID

A test validating the correctness of the SetStaffID and GetStaffID functions in the Librarian class.

Sets the staff ID of the Librarian using librarian.SetStaffID(1). With the Use of the REQUIRE macro there’s a check to see if the retrieved staff ID using librarian.GetStaffID() is 1.

Version Control:

When it comes to version control, the git system is used to track and commit changes and ultimately can push any new additions to an online repository through these commits. The online platform used for this project was GitHub, which utilises the git software, and makes the project easily shareable, typically for collaborations but for this project only to be viewed by others.

The repository is public for easy viewing and contains the source code for the whole project, separate files such as the pdf presentation, video demonstration of the code, and finally, this project report.

Conclusion:

The discussed project is a C++ implementation of an extensive library management system. Important functions include reading book details from a CSV file, creating and displaying maintained librarian details, issued and returned books, and collected member information.

Git/GitHub are the only tools used for version control; yet, early in the development process, additional commits could have been made to improve it from a development process.

To methodically verify features like CSV file reading, member creation, book issuance, and librarian information management, the testing methodology uses Catch2 test cases. Additional test cases for input validation and the implementation of tests that concentrate more on data changes for instance, tracking new members—than on the product could be improvements.

The project features a modular, well-organized design that guarantees efficient management and easy access to library materials.