**CST2550 Coursework 1**

**Erik Vasquez Chafla**

**M00866654**

For this coursework I was asked to create a system that tracks the details of their available books and members in C++. I am going to break down what I did in my code in this report, starting from the beginning in the main file.

A welcome message and a choice to activate the library system (y/n) are displayed to the librarian at the beginning of the program. The librarian is then prompted to enter the file containing book data if they decide to start (y).

A computer screen shot of text

Description automatically generated

To read the book data, the software tries to open the file that is provided through an ifstream. It gives the librarian feedback by telling them if the file can be opened or not. Once a valid file that can be read is inputted the program will then go into a do-while loop, giving the librarian several options to choose from multiple times until they choose to close the program themselves. The librarian is shown a menu of choices within the loop: Add a member (1), See the number of members (2), Issue a book to a member (3), Display all books in the system (4), Close the program (5)

A computer screen shot of text

Description automatically generated

The program uses a switch case to carry out the corresponding function based on the librarian's choice. A member can be added, the number of members can be shown, a book can be given to a member, all books can be shown, or the program can be closed. The switch cases call several functions: AddMember, DisplayMembers, DisplayChosenBooks, and DisplayAllBooks. These functions are the action that will be carried out based on what the librarian choice was.

A screen shot of a computer program

Description automatically generated

At the very bottom of my C++ code is Input validation that makes sure the librarian enters appropriate options. The librarian is prompted to enter Y or N until a valid input is received if an invalid choice is entered. The program displays a message and terminates if the librarian decides not to start the system (n). To handle invalid entries until a valid y or n is supplied, the program a while loop that states that while the input is not n or y then the system will continue to prompt the user.

A screen shot of a computer program

Description automatically generated

All the functions are functions that are in separate C++ files. The functions DisplayAllBooks and DisplayChosenBook, are in charge of providing details about every book in the library and issuing specific books to a member.

To display information about each book, the DisplayAllBooks function reads the contents of the file using the entered file name as an argument (EnterFile). It opens the file with ifstream and uses getline to read the book's details line by line. Book ID, Book Name, Page Count, First and Last Names of the Author, and Book Type are among the book details. In getline, after the datatype that the code is looking for there is ‘,’ at the end of almost every line. This is to separate the data type by using commas. The function prints the formatted contents of each book after looping through the entire file.

A screenshot of a computer program

Description automatically generated

Along with a target book ID (targetBookId), a vector of current members (existmembers), and the entered file name (EnterFile), the DisplayChosenBooks function requires these parameters. It reads the book details from the file and determines whether the target ID (The book ID the librarian is looking for) and the actual book ID match, just like DisplayAllBooks does. In the case that a match is found, the user is prompted to input the Member ID of the member that the book is being issued to. The function checks that the Member ID entered is a valid number and not 0 or a negative by validating it. Next, it searches the existmembers vector to see if the Member ID that was entered already exists in the system. A success message is shown, and the book is added to the member's BorrowedBooks vector if the Member ID is located. If not, an error message appears.

A screenshot of a computer program

Description automatically generated

The process of adding new users to the library system is handled by the AddMember function. To store information about the new member, it initialises a Person object called NewMember. The feature asks the librarian to provide the new member with a Member ID and verifies that it hasn't been used before. If so, a new ID is requested. After that, the function gathers the new member's contact details, such as name, address, and email. The NewMember.BorrowedBooks vector is initialised to contain an empty set of borrowed books for the new member. The feature shows the new member's information, such as their name, address, email address, and any books they have checked out. The vector of current members (existingMembers) gains the new member.

A screenshot of a computer program

Description automatically generated

The library system's DisplayMembers function provides details about each member. The details for each member, including Member ID, name, address, email, and borrowed book IDs (if any), are displayed as iterates through the existingMembers vector. It verifies whether any members have checked out any books. If they have, it shows the book IDs; if not, it indicates that the member has no books checked out.

A computer screen shot of a program code

Description automatically generated

I used classes which have datatype of Person, member and librarian. Which they are all similar but the Person class. The person class has the following datatype: Name, Address, Email, MemberID, BorrowedBooks.

Additionally, the Person class has a member function called BorrowBook that lets someone check out a book by adding its ID to the vector called BorrowedBooks.

A screen shot of a computer program

Description automatically generated