```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <ctime>

using namespace std;

class Book {
public:
   int bookID;
   string title;
   string author;
   string genre;
   bool isAvailable;

   Book(int id, string t, string a, string g) : bookID(id), title(t), author(a), genre(g),
isAvailable(true) {}

   void displayBook() const {
      cout << "ID: " << bookID << ", Title: " << title << ", Author: " << author
         << ", Genre: " << genre << ", Status: " << (isAvailable ? "Available" : "Borrowed") <<
endl;
   }
};

class Member {
```

```
public:
    int memberID;
    string name;
    string address;
    string phoneNumber;

    Member(int id, string n, string addr, string phone) : memberID(id), name(n),
address(addr), phoneNumber(phone) {}

    void displayMember() const {
        cout << "ID: " << memberID << ", Name: " << name << ", Address: " << address << ",
Phone: " << phoneNumber << endl;
    }
};

class Transaction {
public:
    int transactionID;
    int memberID;
    int bookID;
    time_t borrowDate;
    time_t returnDate;
    bool isReturned;

    Transaction(int tid, int mid, int bid, time_t bDate)
        : transactionID(tid), memberID(mid), bookID(bid), borrowDate(bDate),
isReturned(false) {}
```

```
    void returnBook(time_t rDate) {
        isReturned = true;
        returnDate = rDate;
    }

    int calculateOverdueFee() const {
        if (!isReturned) return 0;
        const int dailyFee = 1;  // $1 per day
        int daysOverdue = difftime(returnDate, borrowDate) / (60 * 60 * 24) - 14;  // 14-day period
        return (daysOverdue > 0) ? daysOverdue * dailyFee : 0;
    }
};

// Function prototypes
void addBook(vector<Book>& books);
void addMember(vector<Member>& members);
void borrowBook(vector<Book>& books, vector<Transaction>& transactions, vector<Member>& members);
void returnBook(vector<Book>& books, vector<Transaction>& transactions);
void displayBooks(const vector<Book>& books);
void displayMembers(const vector<Member>& members);
void saveToFile(const vector<Book>& books, const vector<Member>& members, const vector<Transaction>& transactions);
void loadFromFile(vector<Book>& books, vector<Member>& members, vector<Transaction>& transactions);
```

```
// Main menu and functions
int main() {
   vector<Book> books;
   vector<Member> members;
   vector<Transaction> transactions;
   loadFromFile(books, members, transactions);

   int choice;
   do {
      cout << "\nLibrary Management System\n";
      cout << "1. Add Book\n2. Add Member\n3. Borrow Book\n4. Return Book\n5. Display Books\n6. Display Members\n0. Exit\n";
      cout << "Enter your choice: ";
      cin >> choice;

      switch (choice) {
         case 1:
            addBook(books);
            break;
         case 2:
            addMember(members);
            break;
         case 3:
            borrowBook(books, transactions, members);
            break;
```

```
            case 4:
                returnBook(books, transactions);
                break;
            case 5:
                displayBooks(books);
                break;
            case 6:
                displayMembers(members);
                break;
            case 0:
                saveToFile(books, members, transactions);
                cout << "Exiting...\n";
                break;
            default:
                cout << "Invalid choice, try again.\n";
        }
    } while (choice != 0);

    return 0;
}

// Implement add, borrow, return, display, and file handling functions here
```